REPORT DOCUMENTATION PAGE

Form Approved - OMB No. 0704-0188

1. AGENCY USE ONLY (Leave blank)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	August 1996	Technical Report, 7/1/95 - 6/30/96

4. TITLE AND SUBTITLE

ADAPTIVE DECISION MAKING AND COORDINATION IN VARIABLE STRUCTURE ORGANIZATIONS

5. FUNDING NUMBERS

N00014-93-1-0912

6. AUTHOR(S)

Alexander H. Levis

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Center of Excellence in Command, Control, Communications and Intelligence
George Mason University
Fairfax, Virginia 22030

8. PERFORMING ORGANIZATION REPORT NUMBER

GMU/C3I-176-IR

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

Office of Naval Research
800 North Quincy Street
Arlington, VA 22217-5660

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION/AVAILABILITY STATEMENT

unlimited

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

Progress in research on coordination in distributed decision making organizations with variable structure is reported. First results on distributed process coordination in adaptive command and control teams are described. In addition, first results in formulating the problem of modeling the decision making process in a hierarchical C2 organization using influence diagrams and Petri Nets are given.

14. SUBJECT TERMS

Decision Making; Organization Theory; Colored Petri Nets; Influence Diagrams

15. NUMBER OF PAGES

53

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	UL

CENTER OF EXCELLENCE IN
COMMAND, CONTROL, COMMUNICATIONS AND INTELLIGENCE

GEORGE MASON UNIVERSITY
Fairfax, Virginia 22030

ANNUAL TECHNICAL REPORT

for the period

1 July 1995 - 30 June 1996

for

ADAPTIVE DECISION MAKING AND COORDINATION IN VARIABLE STRUCTURE ORGANIZATIONS

Grant Number N00014-93-1-0912

Submitted to
Dr. W. S. Vaughan, Jr. (3 copies)
Office of Naval Research
800 North Quincy Street
Arlington, Virginia 22217-5000

Copies to:
Director, Naval Research Laboratory
Administrative Grants Office, ONR
DefenseTechnical Information Center

Submitted by:

Alexander H. Levis
Principal Investigator

August, 1996

GMU/C3I-176-IR

TABLE OF CONTENTS

1. PROGRAM OBJECTIVES

The objective of this research, as described in the proposal and the previous progress reports, is the investigation of several issues related to coordination in organizations. In particular, an organization is coordinated through direct and indirect means. The direct means includes the set of decision rules that the organization members use and the commands that they issue to each other. Indirect means include the dissemination of information within the organization; for example, organization members may share information or they may inform each other as to the actions they plan to take or decisions they have made. Coordination becomes a complex issue in variable structure organizations. Not only do the decision rules and the information architecture have to work for each fixed structure, but the designer has to deal with the problem, a metaproblem, of coordinating the variability. This becomes a particularly difficult problem in organizations that exhibit substantial complexity and redundancy in their information structure. The redundancy is necessary both for robustness and for flexibility and reconfigurability. In order to address these problems two main tasks were defined; they are described in the next section. In addition, some basic work in algorithms and Colored Petri Nets needs to be done to develop tools and techniques for supporting the analysis and design.

2. STATEMENT OF WORK

The statement of work, as outlined in the proposal, is given below.

Task 1: Consistency and Completeness in Distributed Decision Making

Develop a methodology for analyzing and correcting the set of decision rules used by an organization with distributed decision making. The methodology is to be based on the modeling of the set of decision rules in the form of a Colored Petri Net and on the analysis of the net using S-invariant and Occurrence graphs. The ability to verify and correct the set of decision rules has direct impact on the extent of coordination needed in an actual organization and the resulting communication load.

Task 2: Variable Structures: Heuristic rules in the Lattice Algorithm Constraints

Develop a methodology for considering additional constraints in the Lattice Algorithm. Such constraints include the degrees of redundancy and complexity at the different processing nodes (to be derived from the DFS algorithm of Andreadakis), the projected response time of the organization, ands some user-specified constraints on connections between decision making units. Develop a procedure for checking the validity of such constraints and incorporate them in the Lattice Algorithm. Generalize the approach to multilevel organizational structures and to variable structures, where variable structures are obtained by folding together different fixed structures. The real focus of the task is to introduce these additional constraints as a way of containing the dimensionality problem inherent in flexibility and reducing the coordination requirements.

Design a symbolic interface for the Lattice algorithm. The interface would have the capability of interpreting natural language inputs entered by the user and will include some symbolic processing. The system will generate the interconnections matrices used as input to the Lattice algorithm. The designer would then use the various tests described in the proposal (such as DFS algorithm) to check the validity of the interconnection constraints and to make required modifications.

Task 3: Information Dissemination

The results of this research will appear in thesis reports and in technical papers to be presented at professional meetings and published in archival journals. In addition, oral presentations will be given periodically as arranged with ONR.

Task 4: Dynamic Task Allocation in Adaptive C2 Architectures

The objective of this task is the application of CAESAR II (Computer-Aided Evaluation of System Architectures) to several problems associated with the organizational design of flexible Command and Control Architectures for Joint Operations in modern littoral warfare. Specifically, the tools and techniques developed for the design of distributed tactical decision making organizations and for designing adaptive information structures for such organizations are embodied in the software suite in CAESAR II. When a well defined organizational task is mapped onto the humans and machines constituting the organization, several problems of inconsistency, redundancy, and ambiguity arise that degrade organizational performance. The results of Task 1 - theory and algorithms - will be applied to this problem. Furthermore, the existence of CAESAR II with the enhancements of Task 2 makes possible the support of model-driven experimentation.

Given the nature of this task and the required close coordination with operational DOD organizations and with other organizations involved in this initiative, extensive travel is anticipated by the principal investigator and senior staff associates with this task.

Task 5: Graphical Representation of C2 Decision Making and Supporting Inference

The objective of this task is to use the rapidly developing field of influence diagrams and Bayesian networks to build graphical representations of decision making and supporting inference (intelligence) processing for command and control organizations. In particular, the emphasis is on distributed command and control organizations that are involved in rapidly evolving tactical situations and are likely to reorganize so as to remain effective. As this is our first effort in this area, our focus for this task will be on the representaiton of decision making and inference processes within command and control. The representation issues that we will adress are distributed, concurrent, asynchronous, and interconnected command and control elements; the evolution of decision making tasks throughout a typicl tactical mission; the exchange of information between elements associated with the evolution of time within a specific mission phase; and the impact of overarching the environmental and threat uncertainties that inhibit the effective paritioning of the command and control elements. This activity will lead to critical new representation ideas in influence diagrams and Bayesian networks, enabling later research that addresses partitionaing algorithms of decision making and inference tasks for the purpose of effective allocation of such tasks to command and control elements.

3. RESEARCH PLAN

The research plan describes the strategy for meeting the program objectives. Specifically the research plan is organized around a series of specific well-defined research tasks that are appropriate for theses at master's and Ph.D. levels. Individual students are assigned to each task under the supervision of the principal investigator. Additional staff from the C3I Center are included in the project whenever there is a specific need for their expertise.

4. STATUS REPORT

The focus of Task 1 is the development of a methodology for analyzing and verifying the set of decision rules used by an organization with distributed decision making. The methodology is based on the modeling of the decision rules in the form of Colored Petri Net and on the analysis of the net using S-invariant properties and Occurrence graphs. The results obtained for the two analyses, when applied to a specific form of decision rules, have been presented in the Ph. D. thesis of A. Zaidi that was submitted as the third semi-annual report (dated January 1995). The key paper from this work has been accepted for publication in *Automatica* and should appear in a few months. In addition, a new algorithm for designing organizations of interacting decision makers has been developed as an alternative to the Lattice Algorithm. This new algorithm is based on genetic algorithms; it was presented in the fifth semi-annual report (GMU/C3I-168-IR, dated January 1996) and has been submitted for publication. The new research direction in this task is the development of models and algorithms for distributed coordination in adaptive command and control teams. Early results from this research effort are is described in Section 4.1.

Task 2 has continued to be a focus of activity during this year. The recoding and revision of CAESAR II, the Computer Aided Evaluation of System Architectures suite of software algorithms and tools, has continued and the capabilities of the system are expanding. The application of CAESAR II is now Task 4 and progress in this task was reported in the last semi-annual report.

Task 3 covers the formal efforts for documenting the results of the research as technical reports, conference papers and journal papers. The record of these efforts is presented in Section 8, Documentation.

Task 4 is new for this year. This task is focused on applying CAESAR II to the Adaptive Architecturs for Command and Control (A2C2) program. The first effort in this task was reported separately in a progress report issued in July 1996. A summary description of the work documented in that report is contained in Section 4.2.

Task 5 is also a new task for this year. The results of the first stage of this effort are presented in Section 4.3.

4.1 DISTRIBUTED PROCESS COORDINATION IN ADAPTIVE COMMAND AND CONTROL TEAMS (Perdu and Levis)

4.1.1 Introduction

A team may be defined as a group of experts, with overlapping areas of expertise, that work cooperatively to solve decision problems. Command and Control (C2) teams are a class of teams which carry out information processing and decision making activities associated with the command and control process. The team members are well trained for the tasks they are expected to carry out. In addition, the decision making process is subject to strong time constraints and takes place in a very uncertain environment.

Teams are set up for the performance of specific tasks. For the execution of a task, different functions are allocated to the team members according to their expertise and the organization exhibits a pattern of interactions between team members. Coordination between team members to ensure the coherence of the distributed processes is critical for the effective operation of a team. Moreover, since teams face a variety of different tasks or problems and since no single organization is optimal for all situations, the team organization needs to be adaptive, that is, dynamically reconfigurable to meet changing demands. Here again, coordination between team members is critical, if smooth switching from one configuration to another is to occur.

Coordination between team members is subject to the constraint that the performance of the organization has to remain within acceptable limits or, in other words, that performance requirements are still satisfied. The violation of one or several performance requirements can be the trigger for setting up a new coordination scheme. Measures of Performance of the organization need to be evaluated so that the coordination scheme can be qualified as acceptable or unacceptable.

Coordination can be performed in a centralized way, as it is the case in flexible manufacturing systems. A central controller can keep track of the state of the system and can adapt the configuration according to the state and the demands from the environment. In C2 teams, because of the strong time constraints, the coordination has to be distributed and be user-initiated. To switch from one configuration to another, team members have to collaborate to identify the need for change, to select the new configuration and to smoothly implement the new selected configuration. This distributed coordination is the focus of this paper.

The problem of coordination should be addressed at the design stage of an adaptive team. Coordination for reconfiguration is not (or is weakly) addressed in current Systems Engineering design methodologies applied to distributed human decision making. At most, they recognize the need to allocate a function to several physical components to ensure some

degree of redundancy and back-up capabilities, but fail to define precisely the conditions under which a back-up has to take place. This is due to the fact that these methodologies provide only a static representation of the system or team structure, and that adaptivity and coordination are dynamic in nature. The allocation of the tasks to the team members and the definition of the interactions between them should be done so that each team member knows what to do only on the basis of the information he has direct access to. Distributed coordination would then take the form of coordination rules used locally by the different team members. This paper presents a methodology for adaptive team design. Based on Colored Petri Nets, this methodology allows (1) to derive the coordination rules by analysis of the structural properties of the Colored Petri Net representation of the team operations, (2) to validate and verify these coordination rules, and (3) to predict team performance by simulation of the Colored Petri Net. An example is used throughout the paper to illustrate this methodology.

4.1.2 Design of Adaptive Teams

The aim of the design of adaptive teams is to define the responsibilities of each team member. The first step is to generate the *Operational Concept* that will drive the design. The Operational Concept specifies what the team is supposed to do, the type of tasks it will carry out, the missions it will execute and how it will do them. An Operational Concept corresponds to the broadest requirements.

The second step of the approach is to perform a *Functional Decomposition.* Functions necessary for the execution of the missions defined in the Operational Concept are identified and further decomposed into subfunctions that need to be performed for the execution of the function. This decomposition process can then be applied further to the subfunctions until each subfunction corresponds to an elementary task that can be allocated to a single team member. The *Functional Architecture* is derived by specifying the data exchanged between subfunctions. It can be represented by a Petri net that shows how the functions interact for the execution of a mission. An example is shown on Figure 1. Inputs from the environment are processed by functions f1 and f2. The output of f1 is processed by function f3, while f4 needs the output of both f1 and f2 to be performed. Function f5 needs the results of functions f3 and f4 to produce the team output.

A Functional Architecture can be constructed for each mission and for each input pattern. The obtained Petri Nets need then to be folded together into a single structure that depicts all the functions and interactions necessary for the execution of the set of missions. The resulting structure is a variable one and is represented by a Colored Petri Net.

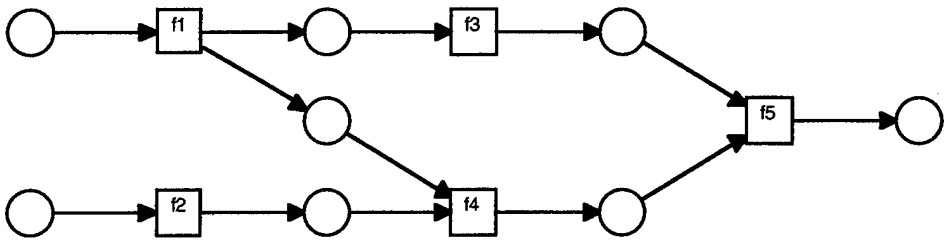


Figure 1 Petri Net Representation of a Functional Architecture

In the last step, functions are allocated to team members subject to the limited processing capabilities of each member, and to the limited capacity of the communication network connecting the different team members. Redundancy of processes and back-up strategies are considered at this stage. Two basic types of back-ups can be considered: vertical and horizontal.

Vertical Back-up

Vertical back-up is the transfer or devolution of responsibilities from one team member to another when the former is unable to perform the function(s) he is responsible for. Vertical back-up is implemented by defining different allocation of functions to the team members for different modes of operation. Figure 2 shows different function allocations to team members for the functional architecture of Figure 1. The upper part of the Figure corresponds to the distribution of responsibilities in a normal mode of operations: DM1 is responsible for function f1, DM2 for function f2, DM3 for function f3, DM4 for function f4, and DM5 for function f5. The lower part is the distribution of responsibilities in a vertical back-up mode when DM5 is unable to perform f5, which is then performed by DM4.

A representation of the team operation is then obtained by folding together the representations of different function allocations. Figure 3 displays the team operation obtained by folding together the two function allocations of Figure 2.

Horizontal Back-up

Horizontal back-up applies to functions that are performed or shared by several team members. When, because of the large number of inputs to process, a function can not be performed by a single decision maker, it can be allocated to several ones so that the overall task is shared between them. Horizontal back-up is therefore an extension of the sharing of the task of one or several team members to compensate for a team member's inability to complete his share satisfactorily. This extension of responsibility is used for load balancing between team members performing the same function. A common way to share responsibilities between team members is to assign geographical areas of responsibilitiy to each of them: a decision maker is

responsible for processing all the inputs in his area. These areas of responsibility can overlap and the team members need to coordinate to decide which input in the common area they each process. Horizontal back-up is implemented by adapting the size of the areas of responsibility. In Figure 4, the functional architecture of Figure 1 has been modified so that the load induced by the execution of function f4 is shared between DM4 and DM5 that have been assigned different but overlapping areas of responsibility. DM1 and DM2 know which input to send to each of them. The inputs in the common area of responsibility are sent to both DM4 and DM5 who need to coordinate, as represented by the places connecting DM4 and DM5.

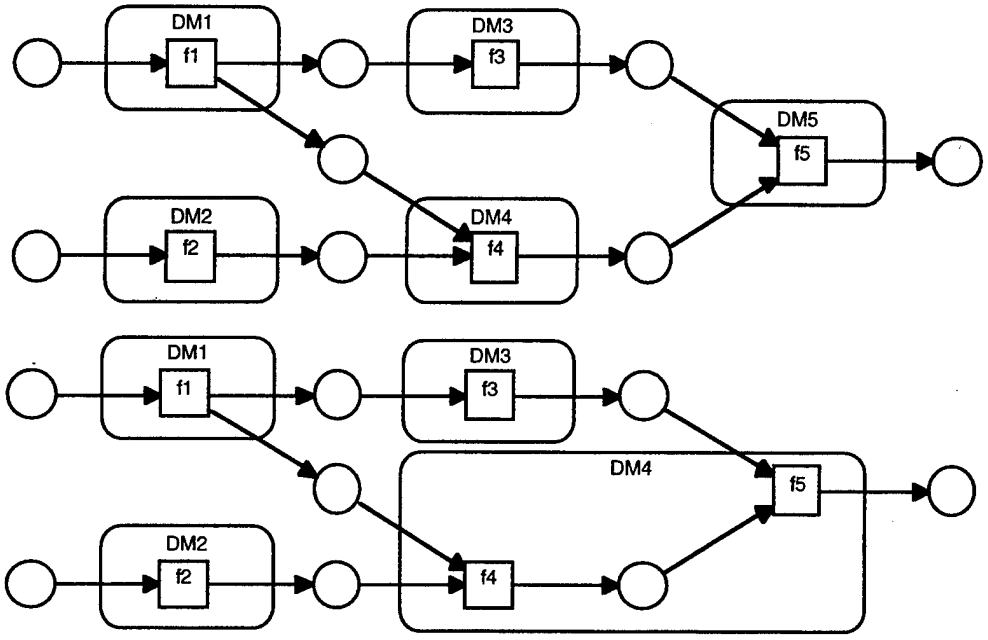


Figure 2 Different Function Allocations for Implementation of Vertical Back-up

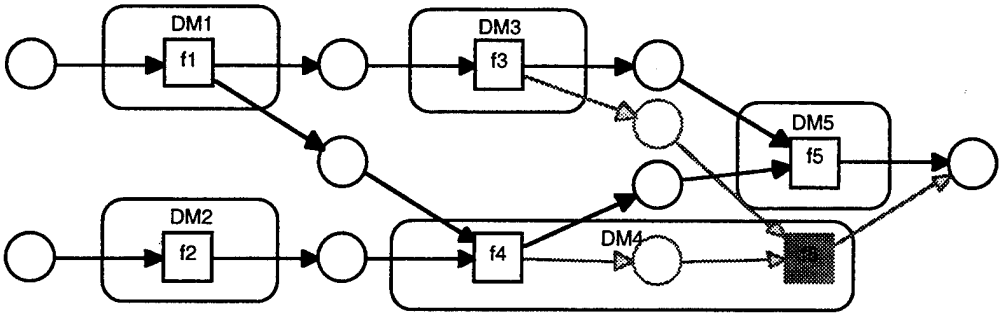


Figure 3 Representation of the Team Operations Accounting for the Vertical Back-up of Function f5 from DM5 to DM4.

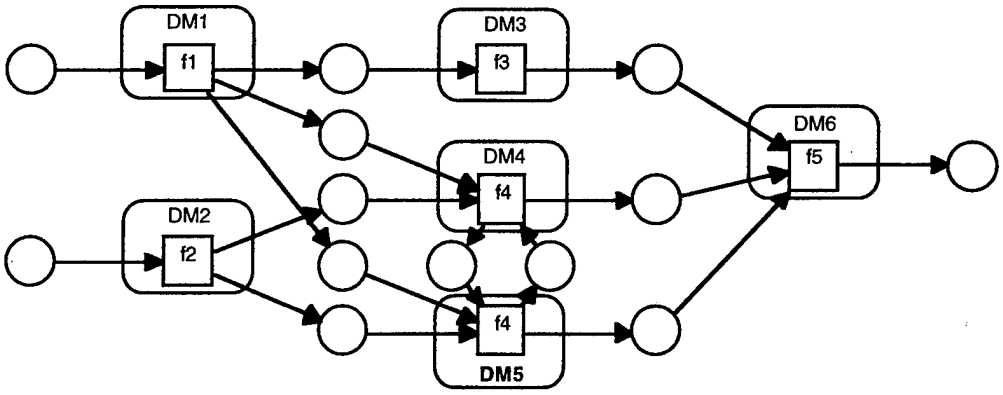


Figure 4 Horizontal Back-up

To specify the different types of operations, it is easier to specify for each function an order of allocation to different team members. In the example, let us consider that the mission has to be carried out by a team of 5 members: DM1, DM2, DM3, DM4 and DM5. The responsibility definition matrix of Table 1 shows the order in which the back-up of each function is implemented. There is no back-up envisioned for f1 and f2, which are respectively performed by DM1 and DM2. The reason is that other team members would need to get the same inputs from the environment to perform this back-up. DM3, DM4 and DM5 are responsible, respectively, for functions f3, f4 and f5 in the normal mode of operation. In a back-up mode, DM1 can perform f3 and DM2 can perform f4. Finally, the back-up of f5 is performed first by DM4 and then by DM3.

Table 1 Responsibility Definition Matrix for the Example

	f1	f2	f3	f4	f5
DM1	1		2		
DM2		1		2	
DM3			1		3
DM4				1	2
DM5					1

A mode of operation is defined as the allocation of each function to each team member. If n functions need to be performed and if the k-th function is allocated to i_k team members in the responsibility definition, there are $i_1 \times i_2 \times \ldots \times i_n$ different possible modes of operations. In the example, the number of different modes of operation is 1 x 1 x 2 x 2 x 3 = 12 modes of operation. The order of back-up implementation for each function introduces a partial ordering

in these modes of operation. The total set of modes of operation can be characterized by a lattice. This lattice can be represented by a Hasse diagram called the mode transition graph. Each node corresponds to a mode of operation. A directed link between two modes indicates that a function has been transferred from one team member to another in the order specified in the responsibility definition matrix. The upper bound is the normal mode of operation, the lower bound is the back-up mode where each function is performed by the last team member in their respective order. Figure 5 shows the mode transition graph for the example.

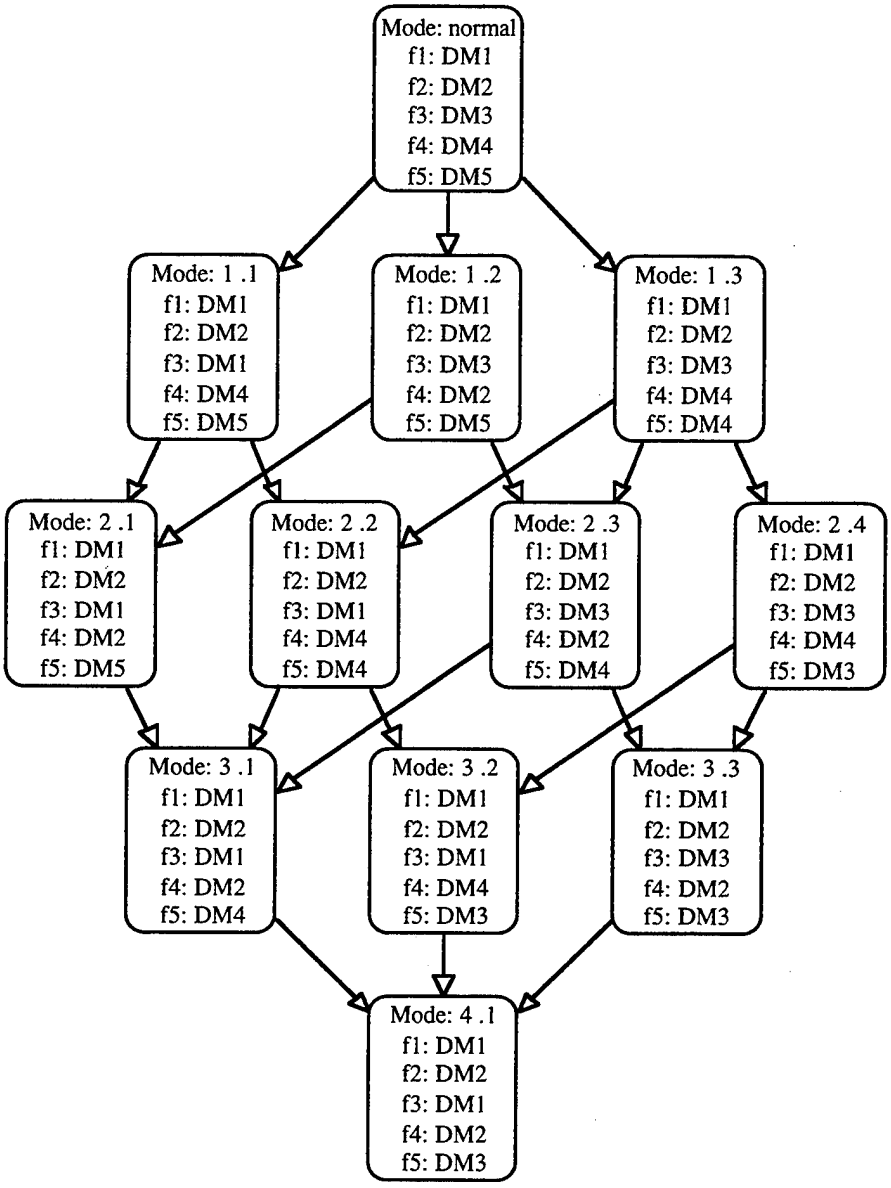


Figure 5 Mode Transition Graph for the Example

To the responsibility definition matrix corresponds a Petri Net, called the fully connected graph, that represents the complete allocation of functions to team members and the exchange of data between team members for the execution of these functions. This net is generated as follows. The different decision makers are represented as rounded boxes. The functions allocated to the team members as defined by the responsibility definition matrix are represented as transitions inside these boxes. Every place of the functional architecture net is replicated as many times as necessary in the fully connected graph to represent the corresponding interaction between team members:

- Source places are replicated as many times as there are instances of the transitions they are the input to,
- Sink places are replicated as many times as there are instances of the transitions they are the output from,
- Interaction places are replicated as many times as there are instances of the pairs of transitions they are connected to. A specific label is defined for each place that specifies the team member sending the corresponding data, the decision maker receiving it, and the functions the place interconnects: 1214 indicates that the data is sent by DM1 to DM2 from function f1 to function f4.

Figure 6 shows the fully connected graph for the example.

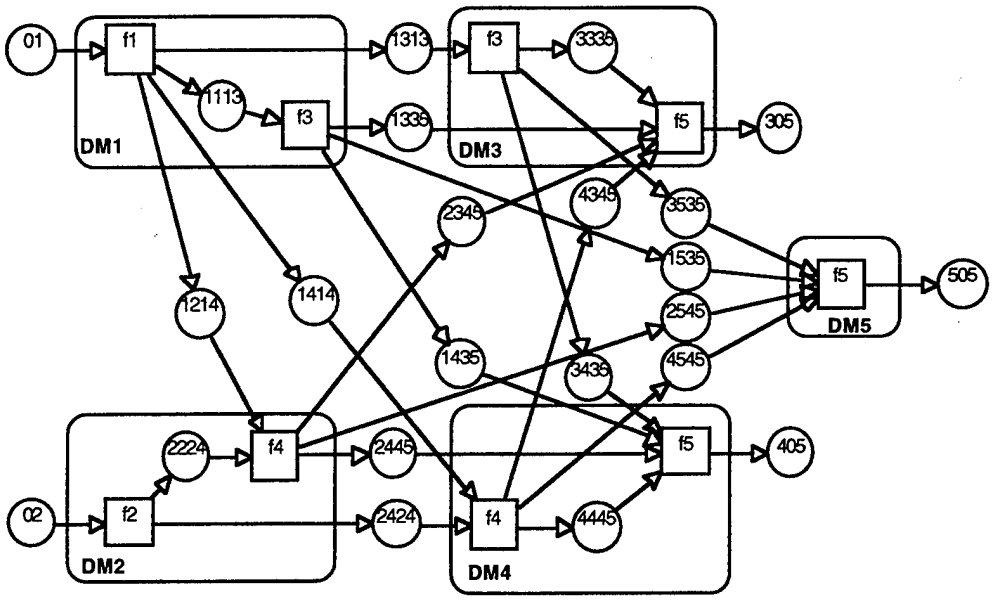


Figure 6 Fully Connected Graph for the Example

The resulting fully connected graph represents all the processes and data exchanges required for the mission to be performed in the different modes of operation. In a given mode of operation, only a subset of transitions and their interconnected places are active. Figures 7 shows in bold the active places and transitions in the normal mode of operation. Figure 8 shows the active places and transitions for the back-up mode where f5 is performed by DM4 (mode 1.3 in Figure 5) and Figure 9 indicates the active links when DM4 performs f5 and f4 is transferred to DM2 (mode 2.3).

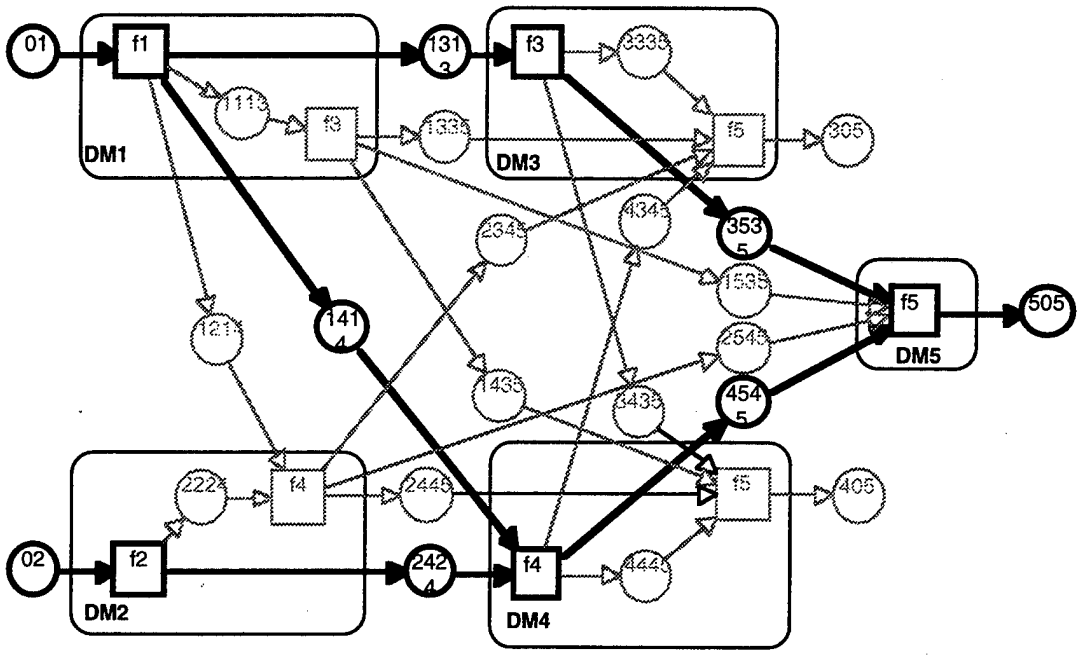


Figure 7 Active Links for the Normal Mode of Operations (Mode 0)

One can see by looking at the Figures 7, 8 and 9 that the transfer of responsibility from some team members to others leads to important changes in the pattern of interactions between team members. There is a need to allocate to team members, along with functions, rules that will help the team members to know under what conditions what functions need to be performed and where the output of these functions needs to be sent. The derivation of these coordination rules is the topic of the next section.

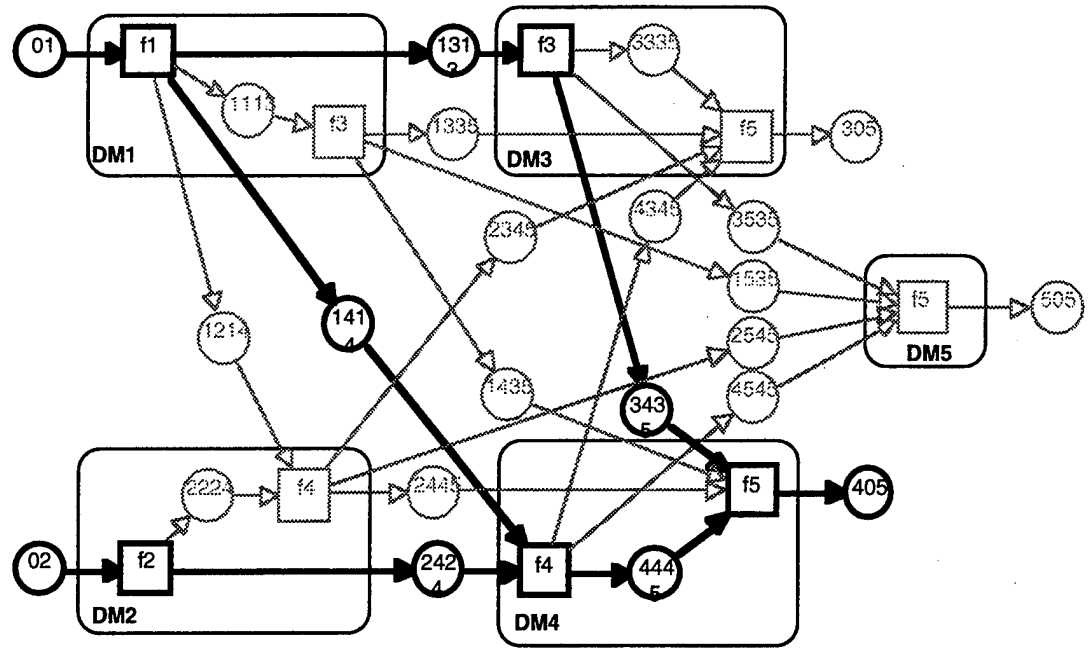


Figure 8 Active Links for Mode 1.3

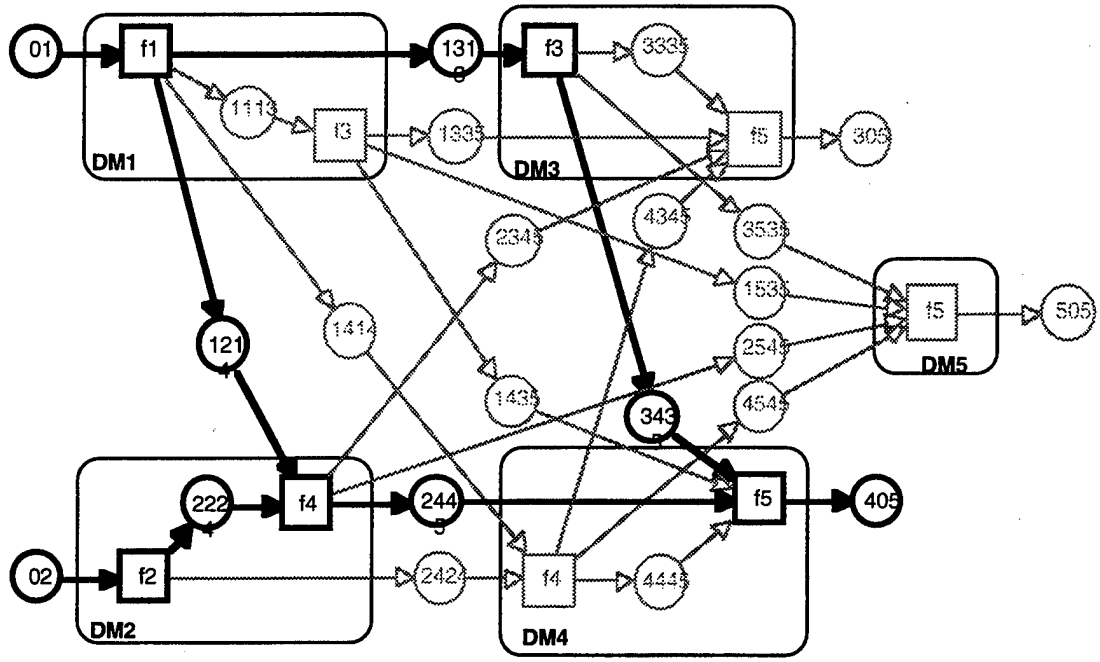


Figure 9 Active Links for Mode 2.3

4.1.3 Derivation of Coordination Rules

Coordination rules need to specify what to do in case an anomaly is detected by a team member. Such anomalies can be (1) the arrival of an item of information not required for the function to be performed in the current mode of operation, (2) the arrival of a message indicating that a problem has occurred and the task can not be performed, (3) the non arrival of an acknowledgment after a certain time-out, or (4) the non arrival of an item of information necessary for the execution of a function.

Central to distributed process coordination is the idea that the team member in charge of performing the back-up of a function needs the different items of information necessary for its execution. To accomplish that, different back-up implementation strategies can be defined:

- Sender Initiated. A team member having performed a function sends the output to another team member who is too overloaded to perform the function in a reasonable delay and "refuses" the message and notifies the sender (or no acknowledgment is sent). At the reception of the refusal message, or when the time-out for reception of the acknowledgment has expired, the sender sends then the item of information to the team member in charge of assuring the back-up of this function as defined in the mode transition graph or rather in its "projection" to the team member.
- Receiver Initiated. The overloaded team member receiving an item of information necessary for a function, sends an acknowledgment to the sender and forwards this item of information to the team member in charge of assuring the back-up of this function. The sender-initiated strategy is still in place for the cases where no acknowledgment is received after a certain time-out.
- Broadcast Strategy. The sender sends the same item of information necessary for the execution of the subsequent function to the different team members that can perform this function in a back-up mode. The team member responsible for the execution of the function in the current mode of operation, receives this information with a specific flag. Back-up is implemented by transferring this flag using either a sender-initiated strategy or a receiver-initiated strategy.
- Other. Other strategies can be constructed, such as for example, the sender keeping track of what he sent to a specific team member and forecasting that the recipient will be overloaded sends the message directly to the team member responsible for the back-up of the function.

Different back-up implementation strategies will result in different sets of rules, adding an other dimension to the problem. The rules have to be generated so that the functions are executed once and only once by the team member. A function for a given task should not be

performed twice (waste of time and resources) and the function should not be non executed because the data necessary for its execution keep going from one team member to another or, because of a "livelock" (Grevet, 1988). A "livelock" occurs when, because of time synchronization problems, different items of information necessary for the execution of the function end up at different team member, each of them waiting for the missing items of information.

The number of different modes of operation is a combinatorial problem that grows with the number of back-ups decided for each function. The derivation of coordination rules could be done manually, but specific attention should be paid to avoid omission or error. When derived manually, the obtained set of rules can be checked with the procedure for validation and verification of a rule base of Zaidi (1994). He developed a Petri Net based approach to identify rules that are redundant, inconsistent or conflicting, circular and incomplete. The use of this method allows to identify the logical problems that can impede the coordination process of adaptive teams when a decentralized approach is used. A systematic approach to derive coordination rules is presented in this section. The described approach generates rules that are allocated to the different team member when the sender-initiated back-up implementation strategy is used. The same approach with some minor modifications can be used for other strategies.

The derivation of the coordination rules is done in two steps. The first step is to define the rules for execution of the functions and the second step is to define the rules for the transition from one mode to another. Execution rules determine which set of inputs to a team member will trigger the execution of a function and where the output should be sent by default. Transition rules determine how the anomalies are generated and handled so that the team member backing-up a function receives the required items of information.

Derivation of execution rules

The functional architecture defines the basic rules of execution. For each transition, the rules are of the type: conjunction of the labels of the input places implies the conjunction of the output places. In Figure 1 representing the functional artchitecture for the example, function f4 requires two items of information: an item produced by function f1 and noted 14 and an item produced by function f2 and noted 24. The execution of this function results in an output required by function f5 and noted 45. The basic rule for function f4 is therefore:

14 & 24 -> 45

The basic rule for function f1 is:

01 -> 13 & 14

where 01 is the input from the environment to function f1 and 13 and 14 are the outputs of function f1 required respectively by functions f3 and f4.

These basic rules are allocated to each team member to which the function has been allocated, regardless of the modes of operation. The rule for f4 is thus allocated to DM2 and DM4, while the rule for f1 is allocated only to DM1.

The triggers for the execution rules are obtained from the fully connected graph and are just a truncation of the label of the places. These rules are allocated to the team member who is responsible for the execution of the function represented by the transition output of this place. For example, the place labeled 1335, input to DM3 to perform f5 leads to the following rule allocated to DM3:

1335 -> 35

To avoid confusion, rules generated so far need to be combined to make the correspondence between the message received and the data necessary for the execution of a function. For example, for function f4, DM4 has the following execution rules:

1414 -> 14 (inputs correspondence)

2424 -> 24

14 & 24 -> 45 (basic execution)

which are combined into:

1414 & 2424 -> 45

For function f5, DM5 has the following rules:

2445 -> 45 (inputs correspondence)

4445 -> 45

1335 -> 35

3435 -> 35

35 & 45 -> 50 (basic execution)

which lead to:

2445 & 1335 -> 50

4445 & 1335 -> 50

2445 & 3435 -> 50

4445 & 3435 -> 50

The next step is to make the correspondence between the output of the basic rules of execution and the message to be sent by default. The default output for each function performed by a team member is determined by looking at the basic modes of operation. The set of basic modes of operatios contains the normal mode of operation and the modes where only one function is backed-up, regardless of the order of the team member doing the back-up. In the example, the basic modes are modes 0, 1.1, 1.2, 1.3 and 2.4. The normal mode helps to

determine the default output of the functions, the other mode when a single function is backed-up. The derived rules establish the correspondence between the output of the derived rules and the message to be sent by the team member. For example, the output of function f4 performed by DM4 in the normal mode is 4545, the rule, allocated to DM4 is therefore:

45 -> 4545

The output of the function f4 performed by DM2 in mode 1.2 is 2545, addressed to DM5 and the rule allocated to DM2 is:

45 -> 2545

The last step is to include the derived correspondence in the rules. The rules for DM4 become:

1414 & 2424 -> 4545
2445 & 1335 -> 405
4445 & 1335 -> 405
2445 & 3435 -> 405
4445 & 3435 -> 405

Derivation of the transition rules

The derivation of transition rules is based on operations on vector representations of the modes. Each mode can be represented by a vector where each cell corresponds to a place of the fully connected net. A cell contains 1 if the corresponding place is active in the mode of operations, 0 if inactive. Table 2 shows the vector representation of the different modes.

Since the rules are local, the restrictions of these vectors to the team members are defined by only considering the places connected to the team member. In the example, the places connected to DM4 are 1414, 2445, 2424, 1435, 3435, 4445, 4545, 4345, 405. By looking at the corresponding columns in Table 2, one can see that DM4 performs exactly the same activities in modes normal denoted by 0 and 1.1.

To derive the transition rules, there is a need to make a distinction between places of a team member that are inputs (places that represent data sent to the team member by another team member), internals (places connecting two functions performed by the team member) and outputs (places representing data sent to another team member) . DM4 has five input place: 1414, 2445, 2424, 1435 and 3435, one internal place: 4445 and three output 4545, 4345 and 405.

Table 2 Vector Representation of the Modes of Operation for the Example

Mode	O1	O2	1113	1214	2224	1414	1313	1335	2445	2424	2345	1435	3335	4345	3435	4445	3535	1535	2545	4545	305	405	505
0	1	1	0	0	0	1	1	0	0	1	0	0	0	0	0	0	1	0	0	1	0	0	1
1.1	1	1	1	0	0	1	0	0	0	1	0	0	0	0	0	0	0	1	0	1	0	0	1
1.2	1	1	0	1	1	0	1	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0	1
1.3	1	1	0	0	0	1	1	0	0	1	0	0	0	0	1	1	0	0	0	0	0	1	0
2.1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	1
2.2	1	1	1	0	0	1	0	0	0	1	0	1	0	0	0	1	0	0	0	0	0	1	0
2.3	1	1	0	1	1	0	1	0	1	0	0	0	0	0	1	0	0	0	0	0	0	1	0
2.4	1	1	0	0	0	1	1	0	0	1	0	0	1	1	0	0	0	0	0	0	1	0	0
3.1	1	1	1	1	1	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	1	0
3.2	1	1	1	0	0	1	0	1	0	1	0	0	0	1	0	0	0	0	0	0	1	0	0
3.3	1	1	0	1	1	0	1	0	0	0	1	0	1	0	0	0	0	0	0	0	1	0	0
4.1	1	1	1	1	1	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	1	0	0

In the next step, for every arc of the mode transition graph, representing a transition from a mode A to a mode B, the vector representation of mode A is subtracted from the vector representation of mode B. The resulting transition vector indicates the changes that have occurred in the transition of mode A to mode B: -1 indicates a link that has to be suppressed, 1 a link that has appeared and 0 a link that remains unchanged. The rules for a specific team member are derived by examining the restriction of the corresponding set of transition vectors. Table 3 shows the transition vectors for DM4. One can see that there are some transition vectors that are identical, others that contain only 0. This shows the indifference of the team member to some modes of operation and the transitions possible from those. The transition vectors address cases when a data sent to another team member has not been received or accepted. Situations dealing with places representing the team output (305, 405 and 505 in the example) are ignored. We are only interested in situations where in the set of output places representing message exchange between team members, the corresponding cells contain 1 or -1. Three different cases can occur.

Case 1: the cells corresponding to the internal places contain 0 and there are one output place containing -1 and another output place containing 1. This case occurs when the function can be performed in back-up by another team member who then needs the data necessary for the execution of this function. If X is the label of the place whose cell contains -1 and Y is the label of the place contains 1, and we note by rX the fact that the item of information X has not been received or accepted, the equivalent rule is:

rX -> Y

This case does not occur in Table 3.

Table 3 Transition Vectors for DM1

Transition	1414 inp	2445 inp	2424 inp	1435 inp	3435 inp	4445 int	4345 out	4545 out	405 out
0->1.1	0	0	0	0	0	0	0	0	0
0->1.2	-1	0	-1	0	0	0	0	-1	0
0->1.3	0	0	0	0	1	1	0	-1	1
1.1->2.1	-1	0	-1	0	0	0	0	-1	0
1.1->2.2	0	0	0	1	0	1	0	-1	1
1.2->2.1	0	0	0	0	0	0	0	0	0
1.2->2.3	0	1	0	0	1	0	0	0	1
1.3->2.2	0	0	0	1	-1	0	0	0	0
1.3->2.3	-1	1	-1	0	0	-1	0	0	0
1.3->2.4	0	0	0	0	-1	-1	1	0	-1
2.1->3.1	0	1	0	1	0	0	0	0	1
2.2->3.1	-1	1	-1	0	0	-1	0	0	0
2.2->3.2	0	0	0	-1	0	-1	1	0	-1
2.3->3.1	0	0	0	1	-1	0	0	0	0
2.3->3.3	0	-1	0	0	-1	0	0	0	-1
2.4->3.2	0	0	0	0	0	0	0	0	0
2.4->3.3	-1	0	-1	0	0	0	-1	0	0
3.1->4.1	0	-1	0	-1	0	0	0	0	-1
3.2->4.1	-1	0	-1	0	0	0	-1	0	0
3.3->4.1	0	0	0	0	0	0	0	0	0

Case 2: There is an internal place whose cell in the transition vector contains 1 and there is an output place whose cell contains -1. This indicates that the function unable to be performed by the recipient has to be performed by the emitter of the message. The same type of rule as in case 1 is generated, the only difference being that Y corresponds to an item of information that the team member needs to keep in memory to perform the back-up of the function.This case occurs for example in Table 3 for the transition of normal mode to mode 1.3: f5 has to be performed by DM4 and not by DM5. The corresponding rule is:

r4545 -> 4445

Case 3: There is an internal place whose cell in the transition vector contains -1 and an output place containing 1. This indicates that the decision maker that has a part of the data from a function previously executed can not perform the subsequent function and has to send the data to the team member responsible for the back-up of the function. This occurs when the team member has already refused to process an item of information. As a result, the team member must remember the messages he has refused. These data are indicated in the input places and the corresponding cells contain -1. If X, Y and Z are the labels respectively of the internal (its cell contains -1), input (its cell contains -1) and output (its cell contains 1) places, the derived rule is:

rY & X -> Z

In Table 3, this case occurs for the transitions from mode 1.3 to 2.4 and from mode 2.2 to 3.2. The corresponding rules are:

r3435 & 4445 -> 4345

r1435 & 4445 -> 4345

This last case introduces the problem of how the refusal messages are generated. The team members have the possibility to refuse a specific message, if overloaded, if there exists a possibility for the back-up of the function. In the example, DM1 can not refuse data necessary for the execution of f3, DM2 can not refuse data necessary for the execution of f4, and DM3 can not refuse data necessary for the execution of f5. These responsibilities have the lowest order in the responsibility definition matrix of Table 1. For the other responsibilities, rules to generate the refusal messages have to be added. Let us consider the case where a team member is responsible for the execution of a function that needs two items of information. Because of the asynchronicity of events, he receives the first item when overloaded and refuses it. The corresponding item of information is then sent to the team member in charge of performing the back-up of this function. After a while, the team member is not overloaded any longer and receives the item. If he accepts it, the two items of information will be located at two different locations in the organization and the function will not be performed. This situation was called "livelock" by Grevet (1988). Specific rules need to be added to avoid this type of situation. If X and Y represent the data necessary for the execution of function F, rX and rY are the refusal message, and noF indicates that the function will not be executed by the team member, the rules to be added are

X & DM overloaded -> rX & noF

Y & DM overloaded -> rY & noF

X & noF -> rX & noF

Y & noF -> rY & noF

These rules can be generated automatically for each item of information represented in the Fully Connected Graph by a place connected to a transition having more than one input. This kind of rules also covers the third case described above.

4.1.4 Team Performance Prediction

The set of rules derived for a particular mission and resulting from a specific allocation of responsibility to team members and a specific back-up implementation strategy need to be validated and compared to other responsibility allocations and strategies. This validation is done by simulating a Colored Petri Net representation of the team operations to predict the performance attained by the team when these coordination rules are in effect. The simulation of this Colored Petri Net representation allows to derive Measures of Performance that can be compared with those obtained when different responsibility allocations and/or back-up implementation strategies have been used, or when no back-up is foreseen.

In order to perform the dynamic validation, a model of the Interacting Decision Maker is introduced. The decision maker model is an "intelligent" one that is modeled as a knowledge rule-based system: the rule base contains the task knowledge of the team member, the decision rules and the coordination rules, the working memory contains deduced facts and data provided by the environment or other team members. The generic model of the team member is presented on Figure 10. Hierarchical Colored Petri Nets have been used so that a model page can be instantiated as many times as necessary. The page model of Figure 10 can therefore be instantiated as many times as there are team members. Color sets using strings have been used so that the model can be used on a variety of problems with only minor modifications. The place "Rule" contains the set of rules allocated to the team member. The place "Data" in the middle of the model represents the working memory of the team member. The transition "infer" applies the rules on the data to generate new facts. When the proper fact has been generated, the team output can be generated when the transition "Produce Output" fires, or the proper message can be sent to another team member when the transition "Send msg" fires. The working memory contains the pieces of information provided by the other team members or the data from the environment when the transition "Receive ext. input" or "receive message" fires. The firing of these transitions is constrained by the number of tasks the team member is currently processing. The place "Ntask" indicates the maximal number of tasks the team member can process simultaneously. The place "ThreatList" indicates the current tasks the team member is processing. The transition "Receive ext input" is enabled and fires only if the number of tasks processed by the team member is less than the maximum, or if the data concerns a task currently being performed by the team member. The transition receive message always fires when a message is received but the expression on the arc connecting this transition to the working memory "data" implements the rule for accepting or refusing a message. If the number of tasks being processed by the team member is larger than the maximum and the received message relates to a new task, the item of information is marked as refused so that the refusal message can be generated when the transition "send msg" fires. If the team member is not overloaded or the received message relates to a task being processed or is a coordination message, the message is accepted.

The model of the team for the example is presented on Figure 11. The transitions DM1, DM2, DM3, DM4 and DM5 are substitution transitions that refer to different instances of the page model of the team member described above. The rules allocated to the different team members are represented in the initial marking of the places "Rule." A rule is represented as a product of two lists of facts and an integer. The first list are the premises of the rules, the second list holds the conclusion of the rules, and the integer is the delay associated with the rule. As an example, the rule representing the execution of f1 by DM1 is:

01 -> 1313 & 1414

and is represented by the marking:

1`(["01"],["1didf1","s13133","s14144","done"],50)

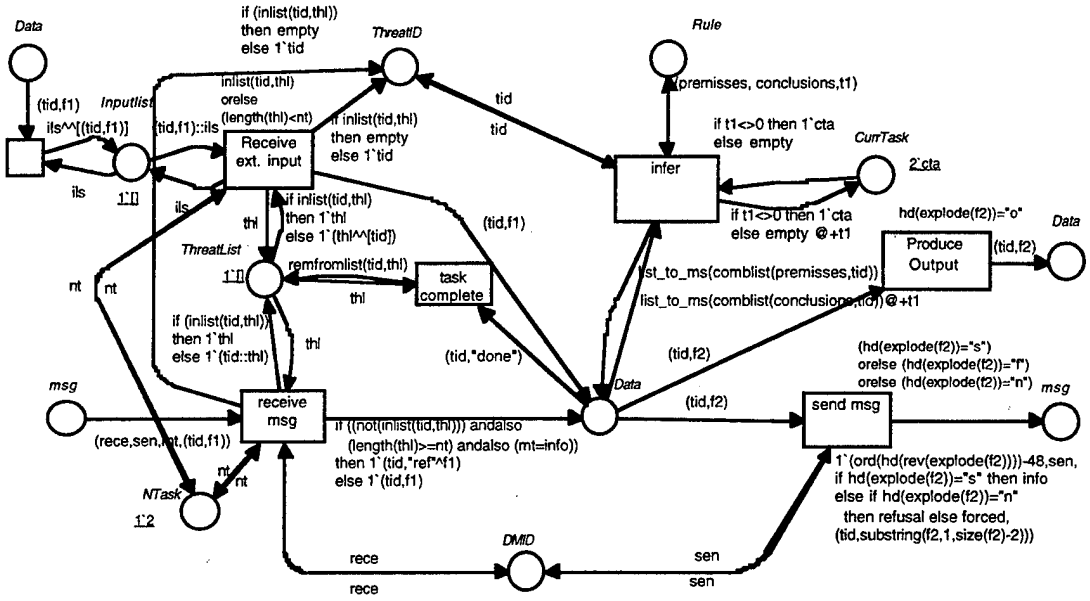


Figure 10 Generic Model of the Team Member

The fact "s13133" is the translation of 1313 to indicate that the item of information 1313 has to be sent to DM3. "1didf1" is to keep track of the activity of DM1. "done" is used to take away from the lists of the tasks being performed by the team member, the number of the task just completed. Finally 50 is the delay for the execution of this rule. The place DMID connected to each substitution transition defines the ID number of each team member and is used for the reception and generation of messages.

The transition scenario generates tokens to DM1 and DM2 that represent data from the environment. The place "ThreatID" defines the ID of the task. The place "Delay" defines the arrival rate of the task. The guard function of the transition scenario indicates the total number of tasks to be processed.

The five team members are connected by a communication network as represented by the substitution transition "Comms." It refers to a model page presented on Figure 12. In Figure 12, messages compete for communications resources represented by tokens contained in the place "CommRes." Coordination messages have higher priority than data messages, require fewer resources and are transmitted after a shorter delay.

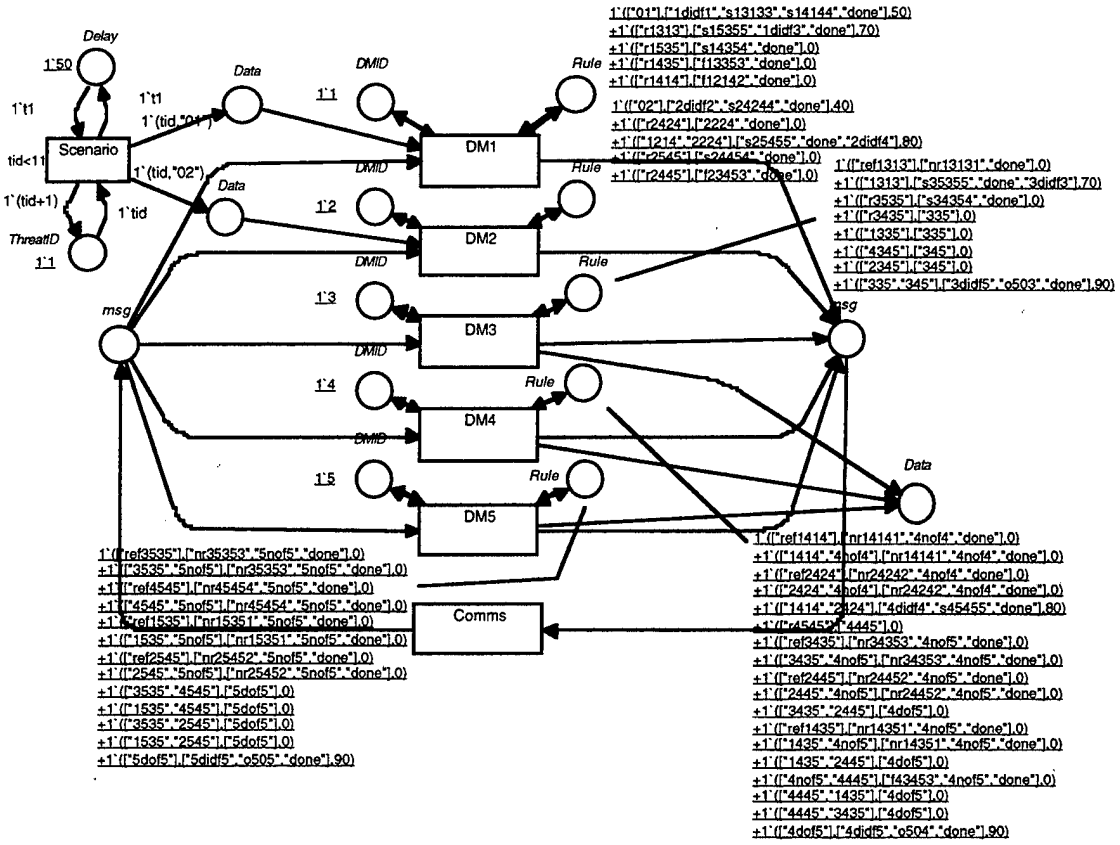


Figure 11 Model of the Team

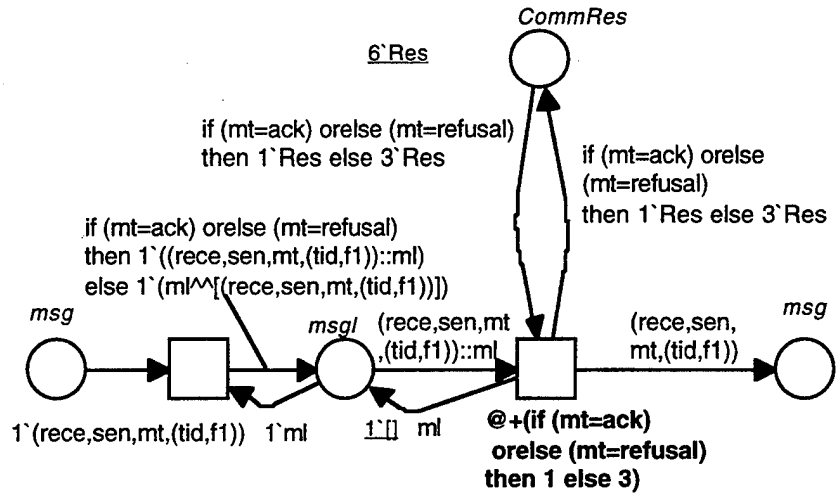


Figure 12 Model of the Communications Network

To evaluate the set of coordination rules, the model of the team operation has been simulated for a scenario of 10 tasks. The arrival rate has been set to three different numbers: low (1 task every 50 units of time), medium (1 task every 25 units of time) and high (1 task

every 10 units of time). Figure 13 shows the average response time to process all the tasks for the adaptive team designed in this paper and for a team with a fixed structure. At a low input arrival rate, the adaptive team does not need to adapt and has the same response time as the team with fixed structure. For a medium input rate, the team with fixed structure is slightly faster than the adaptive team. This can be interpreted by the fact that the transfer of responsibilities between team members is more costly than waiting for the queues to empty. For a high input arrival rate, the adaptive team is more efficient than the fixed one.

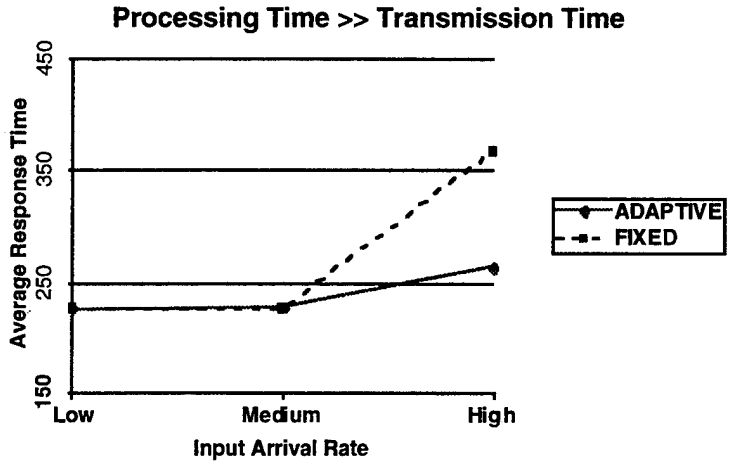


Figure 13 Team Average Response Time

A second set of simulations was carried out to explore the trade-off between implementing back-up and using a fixed structure when the processing time is of the same order of magnitude as the communications time. The results are shown on Figure 14.

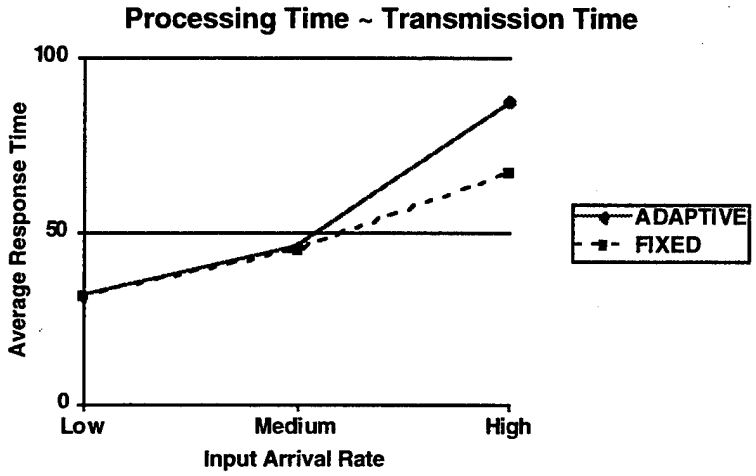


Figure 14 Average Response Time when the Processing Time is of the same Order of Magnitude as the Transmission Time.

The results show that the team with a fixed structure performs better than the adaptive team even for a high input arrival rate. Transfer of responsibilities between team members introduces a lot of communications that can be costly in terms of performance.

4.2 ADAPTIVE ARCHITECTURES FOR COMMAND AND CONTROL (Levis and Perdu)

The A2C2 team developed a wargaming scenario as the basis of the first experiment conducted at the Naval Postgraduate School. The scenario was highly structured and was scripted in detail. At GMU, the scenario was analyzed and the embedded functions were identified. A structured analysis approach was used in the derivation of a discrete event model that realized the given scenario. First, the functions were organized in the form of a functional decomposition structure. This structure was used to develop an activity model in the form of IDEF0. Sevaral iterations were necessary to resolve ambiguities in the scenario. From the IDEF0 model, an executable model was obtained in the form of a Colored Petri Net. The executable model was driven by the scenario variables and two key threads were identified. These key threads were traced in the IDEF0 diagram; a direct form of validating that the model represented indeed the scenario.

The whole process - the reverse engineering of the model from the scenario - highlighted the steps that need to be taken to define fully and unambiguously an architecture. In addition to the activity or process model, a data model needs to be defined that characterizes the information flows in the system; a rule model that contains the rules of the game - under what conditions are certain actions allowed; and a detailed data dictionary. Such a specification, when augmented with a set of state transition diagrams - would provide a rigorous basis for model driven experimentation.

The detailed description of this work - still exploratory in nature - has been documented in a separate report.

4.3 INFLUENCE DIAGRAM REPRESENTATION OF DYNAMIC, DISTRIBUTED DECISION MAKING (Buede and Wagenhals)

This section describes the approach that GMU has taken to accomplish the objectives of Task 5: Graphical Representation of C^2 Decision Making and Supporting Inference. The objective of this task is to use the rapidly developing field of influence diagrams and Bayesian networks to build graphical representations of decision making and supporting inference (intelligence) processing for command and control organizations. In particular, the emphasis is on distributed command and control organizations that are involved in evolving tactical situations and are likely to reorganize adaptively so as to remain effective. The focus for this

task will be on the representation of decision making and inference processes within command and control.

This research is motivated by the large body of research in C^3 architectures dedicated to the generation of feasible distributed decision making organizations based on the formalism of Petri Nets. The mathematical expressiveness of Petri Nets along with the ability to execute them in simulation has facilitated the study of behavior and performance evaluation as the basis for selecting alternative architectures. While Petri Nets provide powerful techniques for analyzing the structure and the dynamics of the distributed decision making organizations, they rely upon an external representation of the decision and inference structure needed to make the C^2 process work. The decision theoretic community of researchers has been expanding formal approaches to modeling decision making processes under uncertainty and making inferences on the basis of incomplete and conflicting data. This research is focused on evolving the decision analysis literature to make it more amenable for incorporation into the Petri Net models.

The representation issues that we will address relate to distributed, concurrent, asynchronous, and interconnected command and control elements; the evolution of decision making tasks throughout a typical tactical mission; the exchange of information among elements associated with the evolution of time within a specific mission phase; and the impact of overarching environmental and threat uncertainties that inhibit the effective partitioning of the command and control elements. This activity will lead to new representation ideas in influence diagrams and Bayesian networks, enabling later research that addresses partitioning algorithms of decision making and inference tasks for the purpose of effective *allocation* of such tasks to command and control elements.

The next section describes the dynamic, distributed C^2 decision making environment as multi-level planning organizations that develop plans, disseminate directives, and monitor progress for revising or changing the plans. The second section defines the basics of influence diagrams, examines previous work that is potentially relevant to dynamic, distributed C2 decision making, provides the basics of our approach, and highlights issues relating to generalization and conversion to Petri Net models of the decision making process. The final section summarizes our accomplishments during this effort.

4.3.1 Overview of Dynamic, Distributed Command & Control Decision Making

Military C^2 systems can be characterized as complex and highly dynamic information systems comprised of decision makers that are organized formally in a hierarchy and whose tasks are distributed over space and time. These decision makers (1) perform many tasks

concurrently, (2) receive inputs asynchronously, and (3) require coordination and synchronization in order to affect the proper sequencing of events. This coordination is accomplished via message passing. The formal organization is hierarchical, but not necessarily the same as that of the functional hierarchy. A generic model of a C^2 system is shown in Figure 1. The military system represented in the diagram is a dynamical system comprised of a set of targets, a set of resources such as those comprising a naval battle force (ships, submarines, planes, etc.), and a command and control system (enclosed in the dark bordered box) to orchestrate its operation to meet the military objective using rules, doctrine, tactics, techniques, and procedures that have formulated and refined over centuries of battles and wars.

The military objective of most combat operations is to attack and destroy a set of targets in the area of operations, represented as the battle space in Figure 15. The target set is not static. Targets move in time and location and are vulnerable to attack during certain time intervals called *windows of opportunity*. The challenge to the C^2 system is to locate and identify targets, determine their window of opportunity, and to get this information to the proper tactical elements so that the correct resources can be brought to bear upon the targets during the time window. This sequence starts with sensor systems that make observations about the battle space. These observations are reported to command centers where they are used by decision makers to assess the situation. The assessment of the situation for both adversarial (red) and friendly (blue) forces is based not only on the observation reports about the battle space, but also on reports from the resource maintainers on the status and availability of the resources at the disposal of the military system. Based on the estimates of the situation, planners devise plans that are comprised of actions that can be taken by the available resources. The Commander evaluates the alternate actions or sets of actions provided by the plans and selects the ones that the he believes will maximize the likelihood of meeting his objectives given the assessment of the current situation and the mix of resources required to implement each plan. These selected actions are passed to the resource preparers so that they know how to prepare the individual resources to be able to execute the actions in the plans. Directives are also provided to the execution controllers who implement the actions by selecting more detailed actions that are given to the resources during plan execution. Finally, directives are sent to those in charge of the sensor systems so that the priorities for information collection are known.

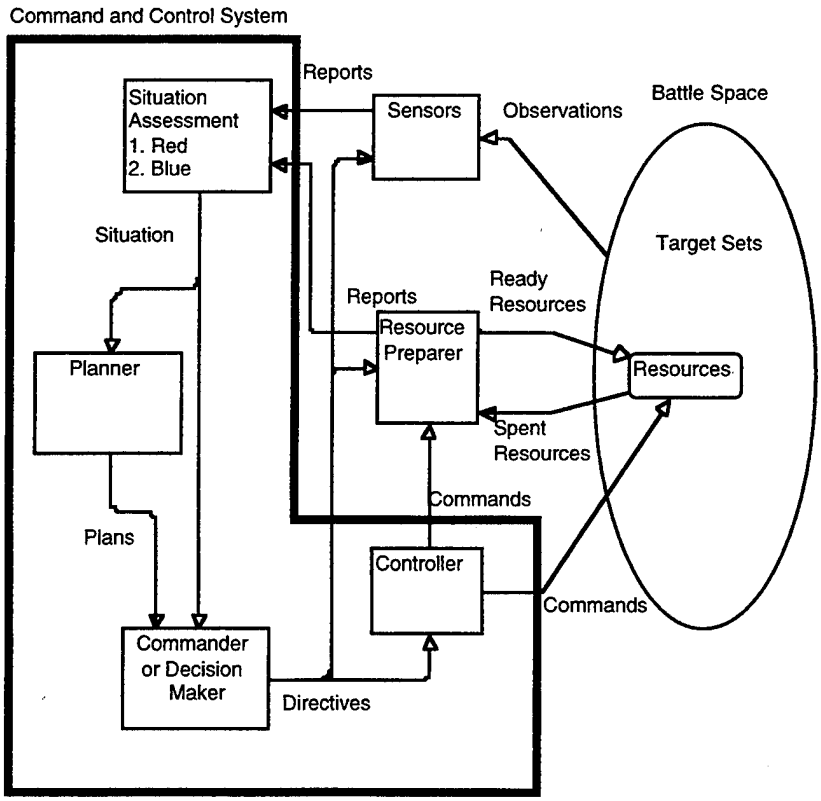


Figure 15. Generic Model of Command and Control System

Most military C^2 systems are more complex then the one depicted in Figure 15. Just as the rules, doctrine, tactics, techniques, and procedures have been formulated and refined over centuries, the structure and operational rules of the C^2 systems have also been standardized. Since the overall task to be performed is much too complex to be handled by a single decision maker, the need for multiple, distributed decision making structures is apparent. This requires a decomposition of the rules, doctrine, tactics, techniques and procedures; the allocation of the components of the decomposition (the tasks) to the distributed decision making nodes shown in the darkened border boxes; and the establishment of the rules of coordination between those nodes in order for the C^2 organization function to meet the overall objectives of the organization. Thus military systems are comprised of several systems that are operating in overlapping areas of the battle space requiring a hierarchical control structure as illustrated in Figure 16 where three hierarchical levels are shown. Level A is the highest level in this depiction. In such systems, higher level command entities receive situation reports from their lower level subordinate command entities. The higher level command entities also draw on a set of plans for the use of resources. Usually these plans are more general than the plans used

by the lower level entities. In this figure, only two controlling elements are shown. In reality many such elements would exist.

Figure 16 illustrates several important features of military C^2 systems. In real military systems, there are groups of specialized resource preparers that prepare their respective classes of resources. For example, in the naval context, there are separate preparers for aircraft, surface ships, and submarines. The resource preparers generally operate concurrently and independently making ready their class of resource with the proper load of weapons, fuel, etc. according to the current plan. This plan specifies when the resources are to be ready for a specified action in the battle space. While the resources may be prepared independently, in many cases they perform their actions in a coordinated fashion once in the battle space. This coordination is facilitated by the controllers. Figure 16 illustrates that there are multiple controllers. In order to accomplish the coordination of the activities of the resources, these controllers must in turn coordinate with each other (not shown in Figure 16) while they send specific commands or directions to the resources they are controlling. Figure 16 also indicates that there are many sensors making observations about the battle space. Different classes of sensors have different capabilities to detect, identify, locate, and determine the activity or state of objects and targets in the battle space. In addition, the higher level decision making organization may have access to sensor information that is not available to lower levels as shown by the sensor box at the top of the diagram. Such sensors may provide a wider view of the battle space than provided by sensors operated by the lower levels of the structure.

As Figure 16 indicates, the activity responsible for assessing the situation does so in two arenas, the situation with regard to the adversary as well as the situation of the friendly resources. This situation is also divided into an estimate of the current situation and a projection of the future situation. Clearly the latter must be based on knowledge of the current plans that have been passed to the resource preparers and controllers. Figure 16 also indicates a partitioning of the planning responsibility. This partitioning is usually along the same lines as the partitioning of the resources preparers. Thus there are planners responsible for aircraft plans, surface ship plans, etc. The generation of these plans is based on the situation and occurs concurrently. This concurrent plan generation can result in potential conflicts between the plans, such as the need for too many resources to be refueled during a limited time window. These conflicts must be identified and adjustments selected to resolve the conflicts. In the architecture depicted in Figure 16, this process is the responsibility of the higher level from its overview vantage point. In other architectures, this conflict detection and resolution process may be distributed between command centers at the same level requiring different coordination paths and rules then those depicted in Figure 16.

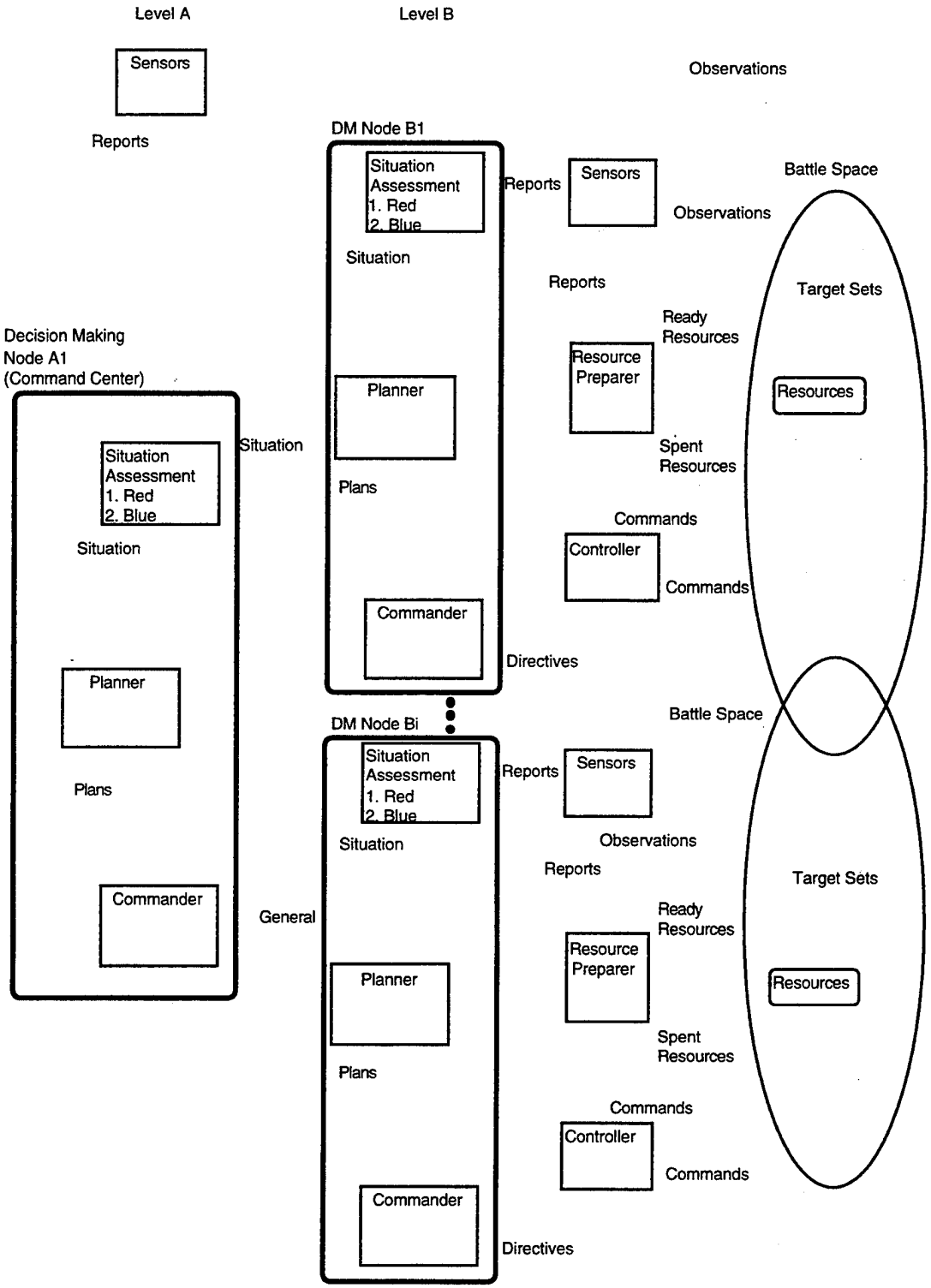


Figure 16 Multi-Level Command and Control Organizational Structure

From this first look at military C^2 systems, it is possible to derive several useful characteristics. Military C^2 systems can be characterized by both their structure and their dynamics. Structurally, they are composed of multiple layers of decision makers that work at increasing levels of generality, the higher one goes in the hierarchy. In general, the higher levels of the structure work on longer term, investment type decisions, that effect the situation in the battle space over a period of hours to days later. The lower levels work on shorter term, operational decisions having more immediate effects. One can view the decision making nodes as being specialized within each layer of the organization. For example, at one layer there may be a decision making node for logistics, situation assessment, and combat operations (strike, anti-submarine warfare, anti-surface warfare, and anti-air warfare in the naval context). Within each node of the distributed decision making organization, the decision problem consists of creating and selecting plans comprised of actions models that are based on the rules, doctrine, tactics, techniques, and procedures. These action models have action/time/space dimensions expressed in the form of a tuple (resource, action, location, time) plus some statements about initial constraints and the expected effect of the action. The action models vary in level of generality depending on the level within the organizational structure. For example, at a high level, an action model might be described in terms of moving a carrier task force to an offensive position 800 nautical miles from were they currently are in a period of one week. Subordinate levels will decompose and specialize the general plan into more detailed actions. At a lower level the detailed action may specify that a particular cruiser should fire a cruise missile at a certain target location at a precise time.

There are several relevant dynamic characteristics of the C^2 organization. The first is that the organization is involved in repetitive decision making based on feedback. Each decision making node chooses a preferred trajectory (plan), observes the actual battle area trajectory, and then chooses corrective actions in terms of revised action models. Thus it is possible to characterize the decision making process as event driven rather than clock driven. With the event driven concept, decisions are made when it is believed that the actual trajectory deviates from the planned trajectory by more than a certain amount or when a new trajectory is directed by a higher level. Another dynamic characteristic is that the decision cycle times vary considerably with organizational level. This is because the higher levels are using more general plans with longer planning horizons than the lower levels.

Clearly, the decisions are being made under conditions of uncertainty since it is unlikely the system can know with certainty the outcome of any action or even the exact current situation. This uncertainty means that at any time there are many possible future states of the world, each with a certain probability of occurrence. Because of the uncertainty, most military C^2 systems develop contingency plans, each with its expected outcome given certain

preconditions. These are available, and thus don't have to be generated, if the situation changes in ways that were not expected. These concepts mean that is unrealistic to model C^2 distributed decision making as a memoryless process. Clearly, the system retains the results of previous decisions, at least for a useful period of time.

A fourth characteristic of military C^2 systems is that they must continually deal with three types of variability. The first is variability in the inputs at each level, the second is the variability of the environment, and the third is the variability of the physical system that performs the distributed decision making functions. Examples of these variations include, inputs varying over time as the situation changes, environmental changes effecting the arrival rate of inputs within the system, and changes in the processing rates of systems within the organization to include failure of systems. To cope with this variability, modern C^2 systems must be adaptable. To cope with this variability, research in the C^3 Architecture design suggests three types of adaptation: change the physical systems and their connectivity, change the coordination schemes, or change the task allocation.

The enemy situation assessment is a very complex process due to the uncertainty and variability described above. This continuous process must accept reports from multiple sensor systems that are received asynchronously. Because the observations are from sensors with a diversity of capability and accuracy, the process must attempt to resolve many ambiguities in the reporting process. Furthermore, it must resolve these reports into hypothesized entities to reduce the tendency to count the same entity multiple times and to enable inferences about entity locations, entity identifications, association with other entities, and entity activities at the time of the observations as well as in the future. Once a current estimate is established, it is necessary to project future estimates.

The friendly situation assessment is less complex than the enemy situation assessment. This is because more reliable, timely and accurate reports will be available to the assessor. Nevertheless, there is still some degree of uncertainty and ambiguity is still possible.

The plan generation process is partitioned among staff decision makers based on a specialization of the process for a given class of resource. Inputs include the situation (enemy and friendly, both current and projected), the current set of plans, and adjustment instructions from the higher level decision maker. In this concept, the planners actually modify existing plans. Certain changes in the situation trigger the plan modification process. In this concept, each decision making node creates new plans or plan modifications asynchronously and concurrently. The organizational design must address the coordination among the planners to avoid conflicts and to synchronize the process.

In order to bound the scope of the research, we propose to concentrate initially on three levels of hierarchical structure of the C^2 distributed decision making structure. We believe that

this will simplify the formulation of the problem without loss of generality to hierarchies with more than three layers. The basic assumption is once the representation of the process has been defined for three layers, it can be expanded to additional layers by recursively applying the process to either the highest or lowest level within the structure in order to add the additional layers. The middle layer will be the focus with the upper and lower layers serving as boundary layers, sending inputs to and receiving outputs from the middle layer.

Before moving on to the discussion of influence diagrams, it is useful to summarize the main characteristics and issues that have been identified from the analysis of the block diagrams.

1. We plan to represent only three levels of the hierarchical structure or the distributed C2 decision making system. The method developed should be easily extended to systems with more layers.
2. The representation of the situation assessment process, will require methods that deal with uncertainty in a manner consistent with decision making.
3. The main decision making process at each level involves the selection of alternative plans. The generation of the feasible alternatives is also a complex problem that will not be addressed in detail in this effort. We will represent the alternative plans as either a generalization/specialization or a composition/decomposition hierarchy. It will be assumed that the alternative generation process exists and creates a set of feasible alternative plans from which the decision maker selects the "best".
4. The decision making process is guided by an overarching set of values dedicated to winning the war. As the situation changes, the decision makers adjust their trade offs among present versus future losses and current losses versus battle gains.

4.3.2 Representing Dynamic, Distributed C^2 Decision Making with Influence Diagrams

4.3.2.1 Overview of Influence Diagrams and Bayesian Networks

Influence diagrams are a graph theoretic representation of a decision. After significant research by Howard [1990], Shachter [1986, 1990] presented the requirements and algorithms needed to transform an influence diagram from solely a communication tool into a computation and analysis tool capable of replacing the standard decision analytic tree. Significant additional research continues into influence diagrams for structuring decision problems, defining the underlying mathematics and graph theory of influence diagrams, and analyzing decision problems. When properly implemented, decision trees and influence diagrams provide

identical solutions to the same problem. They are referred to as isomorphic since the decision tree can be converted to an influence diagram, and vice versa.

An influence diagram may include four types of nodes (decision, chance, value, and deterministic), directed arcs between the nodes, a marginal or conditional probability distribution defined at each chance node, and a mathematical function associated with each decision, value and deterministic node. Each decision node, represented by a box, has a discrete number of states (or decision options) associated with it; chance nodes, represented by an oval, must be discrete random variables. Deterministic nodes are represented by a double oval. A value node may be represented by a rounded cornered box, diamond, hexagon, or octagon.

An arc between two nodes (shown by an arrow) identifies a dependency between the two nodes, see Figure 17. An arc between two chance nodes expresses relevance and indicates the need for a conditional probability distribution. An arc from a decision node into a chance or deterministic node expresses influence and indicates probabilistic or functional (respectively) dependence. An arc from a chance node into a deterministic or value node expresses relevance, that is to say the function in either the deterministic or value node must include the variables on the other ends of the arcs. An arc from any node into a decision node indicates information availability; that is, the states of these nodes are known with certainty when the decision is to be made.

The decision node represents a logical maximum (minimum) operation, that is, choose the option with the maximum (minimum) expected value or utility (cost). A deterministic node can contain any relevant mathematical function of the variables associated with nodes having arcs into the deterministic node. A value node also can contain any mathematical function of the variables with arcs entering the value node. In addition, the mathematical function in the value node defines the risk preference of the decision maker.

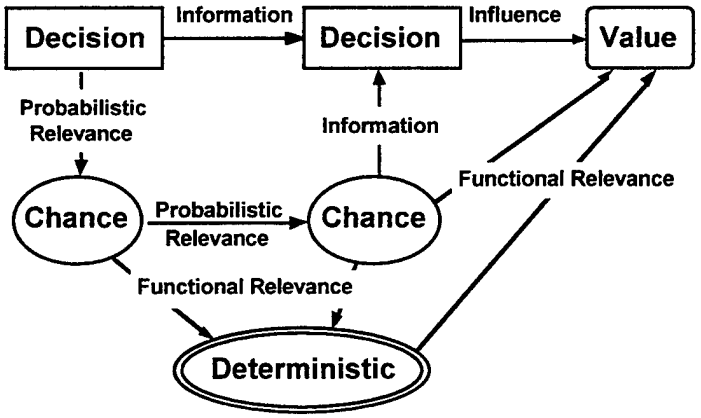
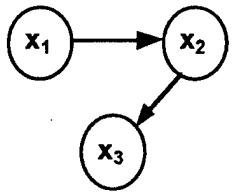
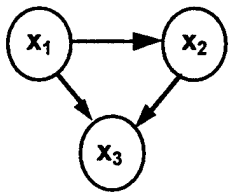


Figure 17. Node and Arc Types in an Influence Diagram

A well-formed influence diagram meets the following conditions: (1) the influence diagram is an acyclic directed graph, that is, it is not possible to start at any node and travel in the direction of the arcs in such a way that one returns to the initial node; (2) each decision or chance node is defined in terms of mutually exclusive and collectively exhaustive states; (3) there is a joint probability distribution that is defined over the chance nodes in the diagram that is consistent with the probabilistic dependence defined by the arcs; (4) there is at least one directed path that begins at the originating or initial decision node, passes through all the other decision nodes, and ends at the value node; (5) there is a proper value function defined at the value node (that is, one that is defined over all the nodes with arcs into the value node); and (6) there are proper functions defined for each deterministic node. An influence diagram that is well formed can be evaluated analytically to determine the optimal decision strategy implied by the structural, functional and numerical definition of the influence diagram.

Decision analysts and Bayesian probabilists have developed another graph-theoretic construct, called Bayesian or belief networks, that is critical to the modeling of C^2 decision making processes. A Bayesian network is (1) a directed graph representing a factorization of a joint probability distribution over n random variables, (2) the joint probability distribution of the n random variables, and (3) a computational architecture for updating the joint probability distribution via Bayes rule as new information is received about any of the random variables. The directed graph contains nodes (one for each random variable) and arcs. The arcs specify which conditional distributions have been chosen in the representation of the joint distribution. For example, the first Bayesian network in Figure 18 has a joint distribution over x_1, x_2, and x_3 that is represented by $p(x_3|x_2)$, $p(x_2|x_1)$, and $p(x_1)$. The absence of an arc from x_1 to x_2 is as important as which arcs are present; the absence of an arc from x_1 to x_3 means that x_3 is conditionally independent of x_1, given the value of x_2. Note, that no joint probability distribution can be represented by a directed graph with a cycle (a directed path that returns to its origin). For a three variable problem there are six possible representations of the joint distribution for the case of full probabilistic dependence; one of which is shown on the right of Figure 18.





$p(x_1, x_2, x_3) = p(x_3|x_2)\, p(x_2|x_1)\, p(x_1)$ $\quad$ $p(x_1, x_2, x_3) = p(x_3|x_2, x_1)\, p(x_2|x_1)\, p(x_1)$

Figure 18. Representations of Two Possible Three-Variable Bayes Nets

In addition to representing a joint probability distribution, a Bayes network provides an inference engine for updating the uncertainty of the joint distribution given new information such as sensor reports. A significant number of probabilistic propagation algorithms have been devised [Pearl, Neapolitan] for networks containing only discrete random variables; the case of normally distributed continuous variables and mixed discrete/continuous variables have also been addressed [Chang] in the literature. The algorithms accept sensor reports, in the form of likelihoods, at any node in the graph as well as changes to the prior distributions at any root node. It is not germane to this discussion to review these algorithms, other than to mention that they (1) can be categorized as exact and simulation-based approximations, (2) are very efficient because they take advantage of the conditional independence present in the graph and (3) are computationally parallelizable. The exact algorithms may be NP-hard for large, highly connected graphs, but this is an initialization issue only.

For multiple sensor and multiple platform applications of data fusion, the sensors may provide inputs to one or more of these nodes. So the Bayesian network is an integration mechanism for the inputs of the sensors; this is the function performed during situation assessment.

An important element of our approach to modeling dynamic, distributed C^2 decision making is to combine Bayesian networks and influence diagrams. Figure 19 shows a simplified influence diagram for a Battle Group command center that has the overall mission of locating and, if necessary, taking military action to destroy enemy systems with a minimum of casualties. There is one fundamental objective - mission success - with three conflicting specific objectives: favorable battle outcome, (max) attrition of enemy forces and (min) attrition of friendly forces. At any point in time the trade-offs amongst objectives can be modified by a higher level C^2 organization, changing the best course of action. The Battle Group's decision is to select the "best" strategy from the defined alternatives. In our simple model only two competing strategies are indicated; in reality there could be many. There is uncertainty about the success of either alternative depending upon the both enemy and friendly current and future status and activity, which includes a number of variables that we will later model as a Bayesian network. This node has been shown as polygon to indicate that it contains the Bayesian network that will be described shortly. Finally, there is uncertainty about whether the friendly forces will attack the enemy forces given the rules-of-engagement and conversely whether the enemy will attack the friendly forces.

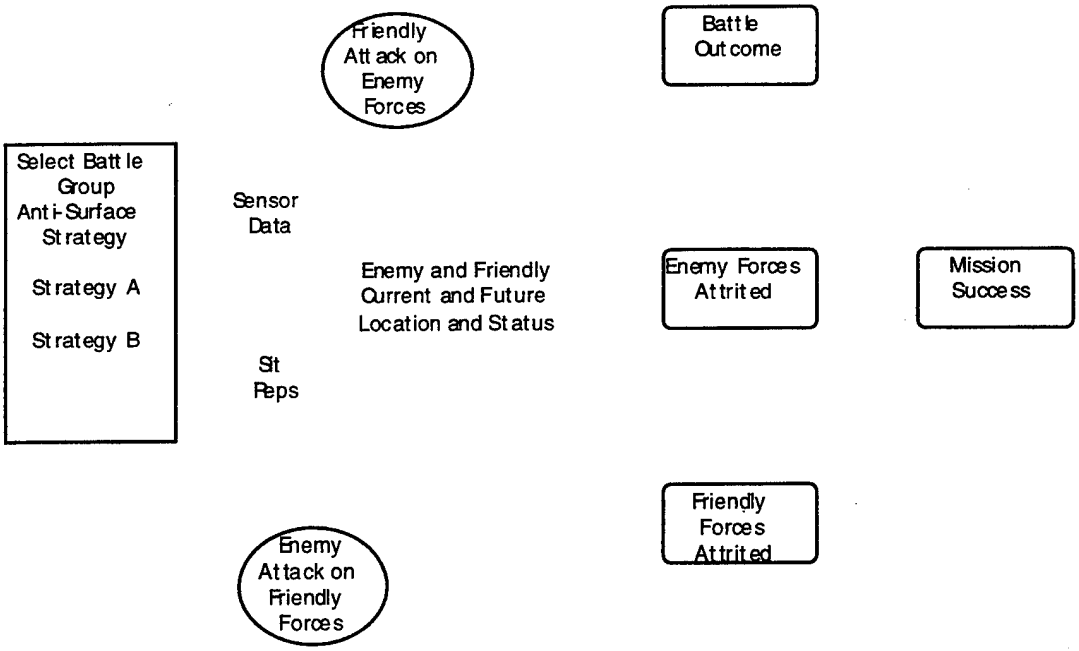


Figure 19. Influence Diagram for Tactical C^2

The influence diagram indicates that the functions that yield the values of the value nodes are dependent on the probabilistic variables indicated in the two chance nodes shown in the diagram plus certain variables in nodes embedded in a Bayesian network, depicted by the irregular shaped and shaded polygon. The chance nodes are conditioned upon variables contained in the polygon and the decision alternative selected. The diagram also shows that sensor and situation reports (octagonal nodes) are also connected to the Bayesian network. There are algorithms that calculate the probabilistic values of the value nodes for each of the possible decision alternatives given the sensor data and situation reports. This means that the decision maker can judge which alternative has the greatest expected value defined by the desired outcome.

4.3.2.2 Previous Work on Dynamic and Distributed Decision Making

Tatman and Shachter (1990) define a generic influence diagram structure (Figure 20) for a multi-period decision process (e.g., dynamic programming). Important elements of this multi-period decision process structure are (1) the segmentation of the decision process into distinct stages in which there are one or more decision nodes, one or more chance nodes and a single value node fed by the decision and chance nodes in that stage only, (2) the informational edges from the chance nodes of that stage for each set of decision nodes in that stage, (3) probabilistic

relevance nodes that cross from one stage to the next stage's chance nodes but no farther, and (4) a final value node that aggregates the value nodes of each stage.

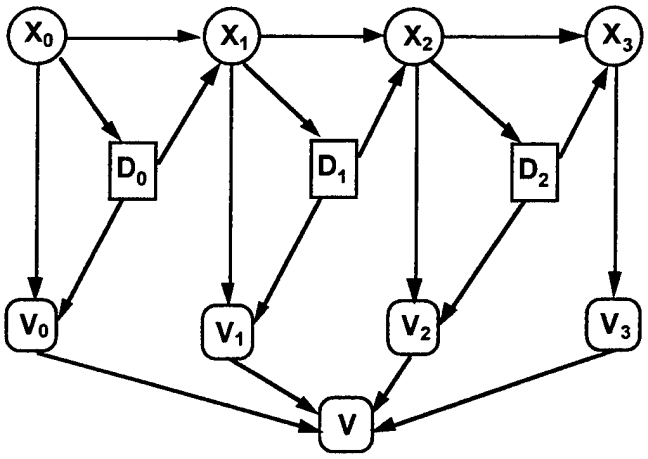


Figure 20. Generic Three Stage Decision Process

Pete, Pattipati and Kleinman (1995 and forthcoming) have presented two approaches for designing organizational structures using the concepts of influence diagrams. Both approaches involve using influence diagrams to model the decision making structure. (Note they choose to drop the concept of value node.) In both cases they constrain the organization design to meeting the assumption that the decision is equivalent to the optimal decision that a centralized decision maker would obtain; they define an organization meeting this constraint as a congruent organization. In both cases they set up an optimization problem of finding the minimum communication network necessary to support the congruent organization in reaching its optimal decision.

One approach [Pete et al., 1995] addresses distributed decision making organizations in a non-dynamic environment that are obtaining situation assessment reports about the environment but are not able to affect this environment based upon their decision options. Such organizations are likely to be information processing organizations, such as an intelligence element. While this work is interesting, it completely ignores the entire literature on Bayesian networks which provide a powerful and consistent approach for examining the situation assessment problem.

The second approach of Pete et al. [forthcoming] addresses distributed, dynamic decision making organizations. Each element of the organization receives imperfect reports (Y in Figure 21) at time t about the environment (H) allocated to it and then makes a decision (D) that impacts the environment allocated to it in time t+1. It is also assumed that the states of each segment of the environment at time t impact the state of every segment of the environment at

time t+1. It is also assumed that at any time t, the decision makers make their decisions independently, but then communicate their decisions to all other decision makers so that this information is available at time t+1. The cost (C) that is being minimized is a function of H and D at any point in time for each decision maker. Total costs can be aggregated for each decision maker and then across decision makers. Note that Pete et al drop these value or cost nodes. Figure 21 shows these assumptions in an influence diagram similar to the one developed by Tatman and Shachter (1990). The results developed by Pete et al. are quite valuable but limited to a single decision making layer, not the hierarchical structure developed above that is typical of C^2 organizations. Also, the situation assessment model representing environmental uncertainty is more neatly structured than is common in hierarchical C^2 organizations. Finally, given the voluminous communications typical of C^2 organizations it is unlikely that such an organization would be interested in minimizing a very small subset of those communications in determining its organizational structure.

4.3.2.3 Dynamic Decision Networks -- Our Approach

Our approach is to use dynamic decision networks (DDN), a set of interconnected influence diagrams and Bayesian networks. Influence diagrams are a decision analytic and graphical construct for representing a decision problem in terms of decision, uncertainties and values, and the probabilistic and informational interactions among them. Bayesian networks are inference engines that model complex interdependent stochastic processes.

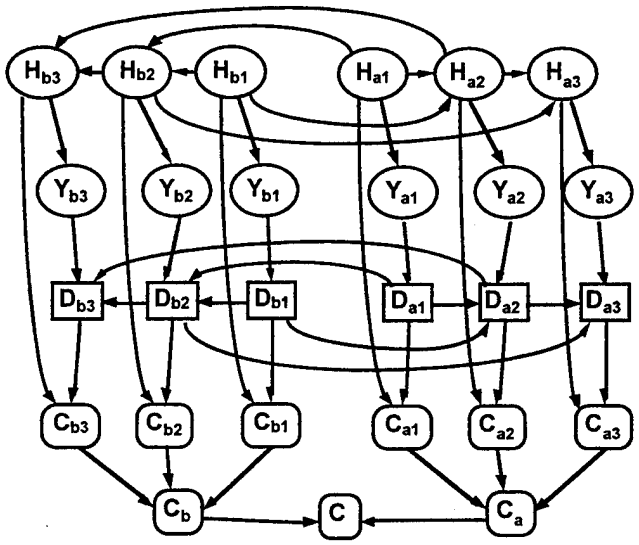


Figure 21. Multiple Decision Maker Network of Pete et al [forthcoming]

There will be a DDN, see Figure 19, for each organizational element of the C^2 organization. We will begin by modeling the decision process of each organizational element of the C^2 organization with a single decision node that addresses the options available to the specific organizational element. For each influence diagram there will be a Bayesian network, see Figure 22, that maintain the current uncertainty on all relevant random variables, based upon sensor and intelligence reports. These chance nodes represent the situation assessment being done by that element. Since the situations for each organizational element are linked to the same battlefield events, the Bayesian networks across organizational elements are likely to be part of one large connected graph.

Figure 22 shows the Enemy and Friendly Current and Future Location and Status polygon node in Figure 19 exploded into the Bayesian network. Several of the nodes in the figure would be updated from sensor and situation reports that are send to the Battle Group command center. (Nodes representing these reports have not been shown in the figure.) For example, the nodes containing the current location the enemy forces (air, surface, and sub-surface) would be updated from sensor reports while the current location of friendly forces would be updated by both sensor reports and situation reports from those forces. Current weather would be updated from weather sensors.

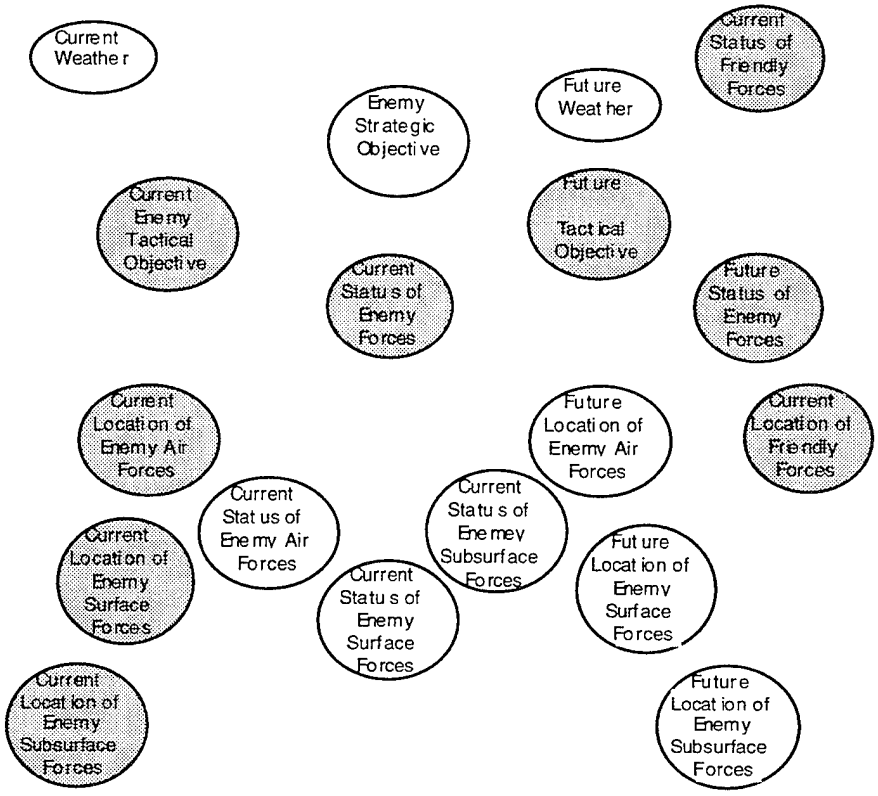


Figure 22. Bayesian Network Embedded in the Tactical C^2 Influence Diagram

To model the distributed decision making process, each decision making node of the C^2 system will comprise a specialized influence diagram similar to the one that was previously shown in Figure19 that represents the decisions to be made at that level. Each node will have a specialized embedded Bayesian net that will be topologically similar to the one shown in Figure 22. A key concept in modeling the distributed decision making process will be determining how to interconnect these specialized influence diagrams with their embedded Bayesian networks. For the Bayesian inference process to work correctly the Bayesian network nodes in each influence diagram at all levels of the decision making process will have to be interconnected; specific C^2 centers will be tasked with updating specific nodes in the Bayesian network and communications amongst the C^2 centers will be needed to maintain the consistency amongst Bayesian networks that are interconnected. The shaded nodes in Figure 22 represent the nodes allocated to the C^2 center represented in Figure 19 for updating.

A partially complete influence diagram of the entire C^2 system can be constructed as shown in Figure 23. For clarity, not all the relevant chance nodes are shown, and many of the vertical and horizontal connections have been suppressed.

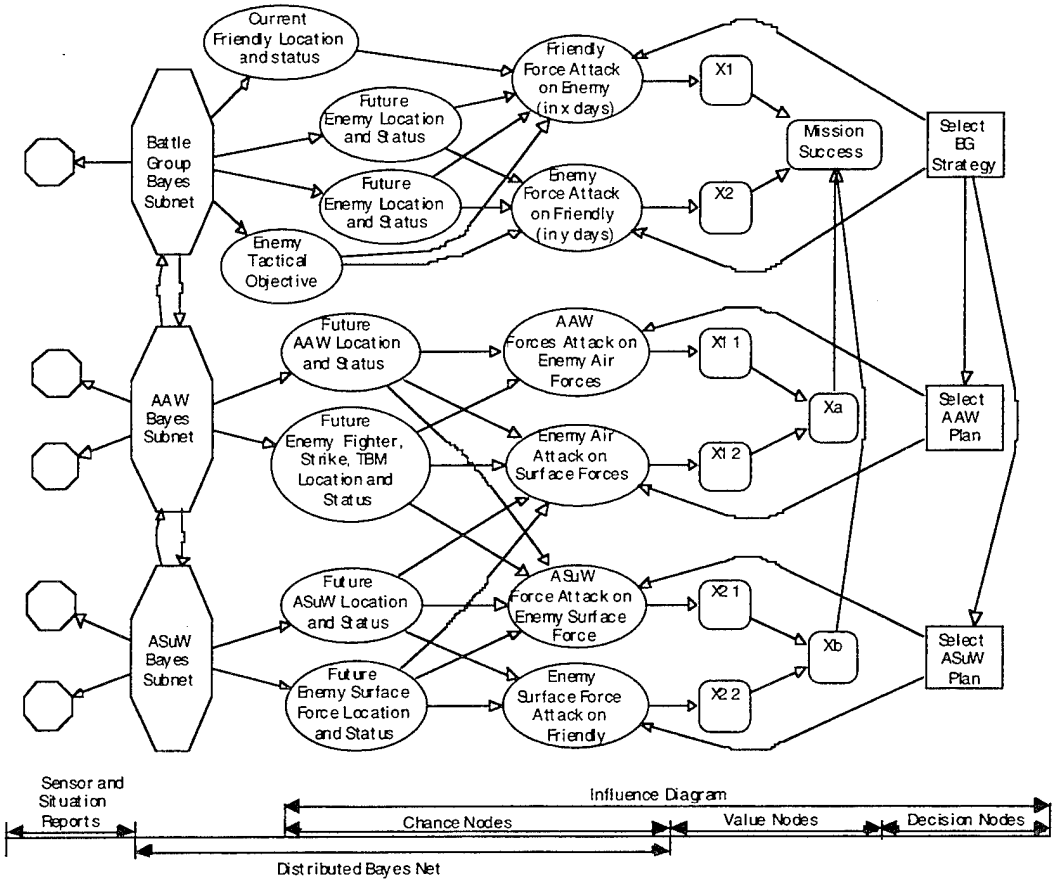


Figure 23. Overall Influence Diagram with Imbedded Bayesian Net

Given such a Bayesian network, the primary responsibility for the nodes must be allocated to the physical elements of the C^2 organization with the needed connectivity. Figure 24 illustrates such an assignment of nodes to the elements of the organizational structure. The arcs that interconnect the elements indicate the communications connectivity required for the data and information passing necessary to operate the Bayesian network algorithm.

Time is represented as a set of event-driven decision making activities. Events outside of the decision making process are modeled by reports received from sensors and sources. When the battle with the enemy is hot (e.g., weapons are being fired) the report events on the order of minutes or seconds for the tactical C^2 organizations that we are considering. When there is no direct contact between forces, a report events might be hours long.

At a given moment in time, we will assume there is directed graph that links the decision nodes in the DDNs of each organizational organization, creating a single DDN for the group of C^2 organizations (Figure 24). This assumption means that we have determined that the organizations will make their decisions in a pre-defined order. This assumption would be very problematic at lower levels of the C^2 process and for operational units that must decide to return fire in seconds or less and cannot wait to find out what other sibling organizations are doing. However, at higher levels of C^2 organizations we can assume that the decision order amongst C^2 elements is directly related to their primacy in facing the current situation, as determined by the commander of the organization being modeled. As the situation changes, this commanding officer can issue directives that change the primacy of subordinate C^2 elements, thus changing the order of the decision nodes in the overall DDN.

4.3.2.4 Conversion to Petri Nets

One of our long-term objectives for this project is to be able to convert the representation of the distributed decision making process of a C^2 organization as a collection of separate influence diagrams into the Petri Net representation that has been used to great advantage in the research on the design of C^3 architectures. Petri Nets in general, and hierarchical colored Petri Nets in particular, have been extremely useful in creating executable models of C^3 systems that can be used in simulation to uncover behavioral characteristics and to explore performance issues where temporal aspects induced by delay in communications paths are important. To refine our description of this aspect of the research problem we need to shift from the block diagram representation technique of Figures 15 and 16 and the graphical representation of the influence diagram of Figures 23 and 24 to that of a less-than-fully-specified (LTFS) Colored Petri Net (CPN) of the intelligent command and control node [Levis, 1992]. The CPN is not completely specified because we have not fully defined the colorset, specified the rules of

execution in terms of arc inscriptions, guard functions, time regions, or code segments, nor established any initial markings. Of course all of these will be necessary to create the executable model.

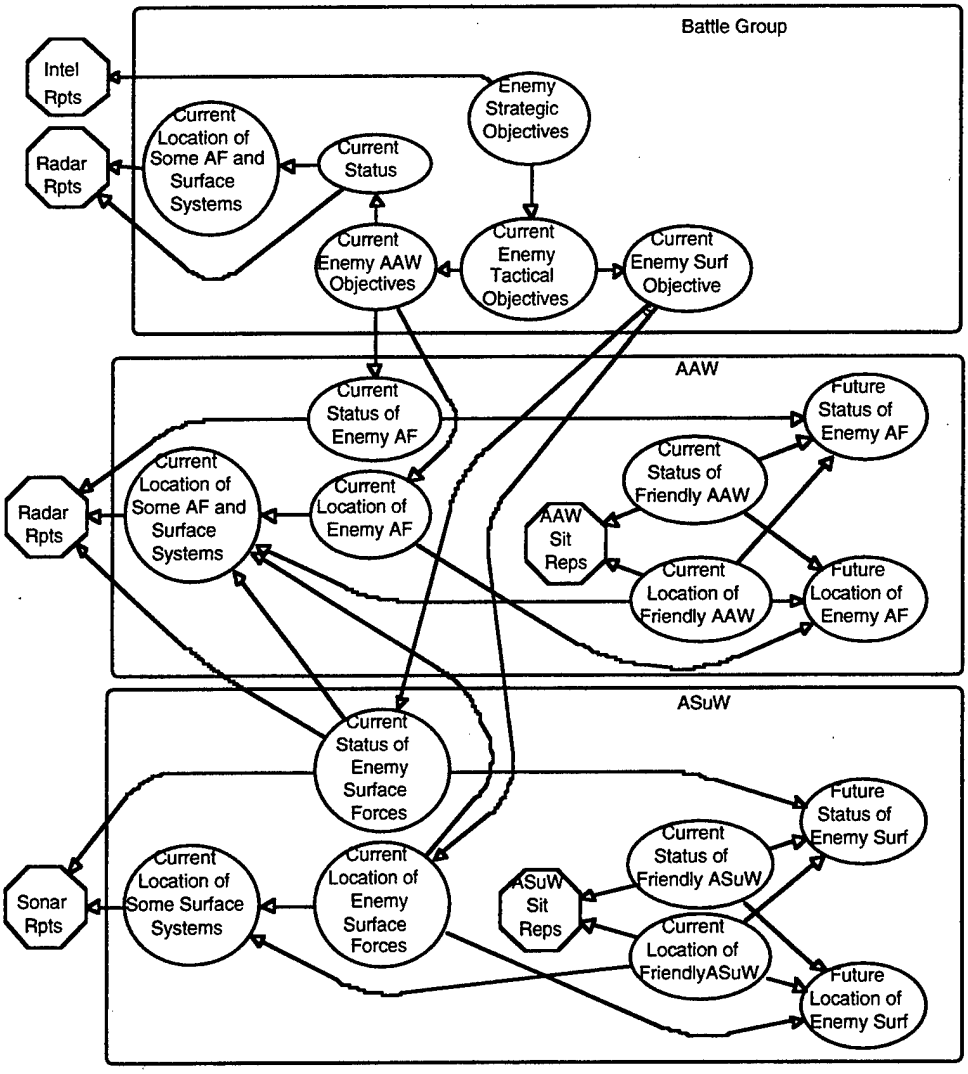


Figure 24. Allocation of Bayesian Network Nodes to Organizational Elements

Once the structure of the decision making process has been defined using the influence diagrams and Bayesian network, it will be converted to a Colored Petri Net executable model using standard techniques that have been developed for modeling C^3 architectures. Figure 24 shows a notional Petri Net model. This Petri Net model is based on the five stage decision making process of Levis (1992): situation assessment, information fusion, task processing, command interpretation, and response selection. The Petri Net structure shown in Figure 11

for each C^2 center has two nets, one for maintenance of the tactical picture based upon our notion of Bayesian networks and one for tactical decision making. These two nets have been separated because their purposes are substantially different and there is no requirement that they run synchronously. Typically the two stage net devoted to the tactical picture will repeat at a faster rate than the second net devoted to decision making; however this can change if needed.

The first net for maintaining the tactical picture only requires the first two of the five stages of decision making. During situation assessment sensor reports from sources under control of the C^2 center are received and processed. These results are output to the information fusion stage of the C^2 center as well as the information fusion stages of upper, lower, and peer C^2 centers. It is during the information fusion stage that updating of the tactical pictures occurs in a manner such that all of the tactical pictures across the C^2 centers are consistent. The results of information fusion at each center are then passed to the decision making process of each center.

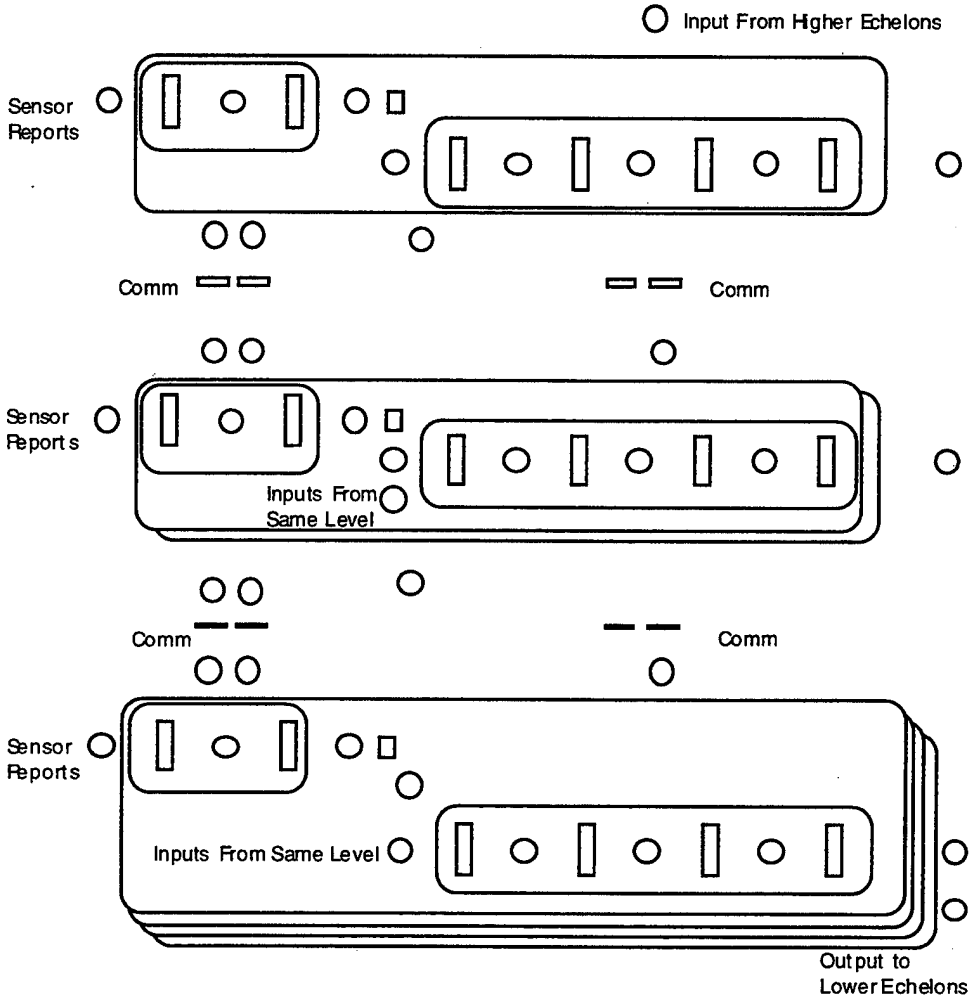


Figure 24. Generic Petri Net Model of Hierarchical Distributed Decision Making System

We have chosen to begin the decision making process with the second stage of the stage model, information fusion. The purpose of this information fusion activity is to integrate the current tactical picture from the other net in the C^2 center with inputs from lower and peer C^2 centers about the status of their decision making operations. The results of this information fusion process are then sent to the task processing stage; here the currently executing plan (decision option) is assessed, new options are generated if desired, and future outcomes are examined for as many options as desired. The focus of these options is to achieve the proper balance amongst resources for lower level elements so that these lower level elements can achieve the results needed for the currently chosen plan. These results are sent to command interpretation where they are compared to the latest guidance that has been received from the higher C^2 node concerning constraints or guidance about feasible options and values on objectives related to the selection of an option. During command interpretation this upper level guidance is used to sift through and understand the impact of the options developed during task processing. The output of command interpretation is then sent to response selection where the final decision is made and documented for communication to upper, peer and lower C^2 centers. It is expected that the time scales for repetition of the decision making process will be faster the lower the C^2 center is in the hierarchical organization. This is a result of the fact that the planning horizon is longest at the highest C^2 center and decreases as we move lower in the C^2 hierarchy. There is no requirement that the planning horizon or repetition interval be the same for C^2 centers that are peers in the hierarchy.

The communication between lower and higher C^2 centers is not only critical to pass guidance and information about the selected plan from above to below. In fact, an equally critical issue in making this distributed, hierarchical decision making organization work in a dynamic environment is to enable the lower level C^2 centers send information describing their difficulties in achieving the desired results so that minor and major adjustments can be made to the selected plan in as timely a manner as possible. A key performance issue relating to decision quality is whether needed adjustments are made to the evolving plan in time to achieve the results needed. The time required to make these adjustments is a second performance issue.

4.3.3 Results of this Work

We have used influence diagrams, which were developed to model decisions of an individual, to examine the decision making process in a hierarchical C^2 organization in which the decision making process is distributed and evolving over time. In order to accommodate these properties of C^2 organizations we have had to introduce the notion of dynamic decision networks, which enable dynamic decision making processes to be simplified for repetitive

decision making tasks. C^2 decision making qualifies as repetitive because a plan is formulated at the highest level and flowed down the hierarchy so that it can be instantiated at lower levels of detail and modified over time as more information is collected about the status and effectiveness of friendly and enemy forces.

By modeling the C^2 decision making process at multiple levels of the organization we have been able to define what information needs to be communicated between levels and among peers at the same level. This information addresses the options, values and facts associated with decision making.

We have formulated a Petri Net model that addresses the key aspects of dynamic decision networks: updating the tactical picture and making timely decisions on the basis of the tactical picture. This Petri Net model is based on the five stage C^2 decision making model developed previously and employs two nets within each C^2 center; the first maintains the tactical picture and the second performs the decision making (plan selection) function. This model is a discrete event model that permits the tactical picture and decision making processes to proceed asynchronously, as they must. Also the repetition cycles of each net within the hierarchy of C^2 centers can vary greatly; lower level C^2 centers are operating at a much greater level of detail and therefore will have to plan at shorter time horizons and repeat more frequently than higher level C^2 centers.

Critical performance issues of C^2 decision making besides the time it takes to initiate a plan is the ability of C^2 organization to detect that the currently selected and executing plan is deviating from the objectives set for it and cannot be modified to achieve acceptable objectives. The Petri Net model we have developed is designed to enable the detection of such a planning problem and enable the communication of needed information as rapidly as possible so that an alternative plan can be selected. The Petri Net model should be used to address performance issues relating to time and communication links needed to detect such problems and how these results change as the C^2 organization adapts its architecture.

4.4. SUMMARY

Progress achieved on all tasks during the year period has been reported.

4.5 REFERENCES

Chang, K.C. (1995) Symbolic Probabilistic Inference with Both Discrete and Continuous Variables, IEEE Transactions on Systems, Man and Cybernetics, Vol. 25, No. 6, pp. 910-916.

Demaël J. and Levis A.H., 1994. On Generating Variable Structure Architecture for Distributed Intelligence Systems. *Information and Decision Technologies*, pp. 233-255, 1994.

Grevet J.-L., 1988. *Decision Aiding and Coordination in Decision Making Organizations* LIDS-TH-1737, MS Thesis, Laboratory for Information and Decision Systems, MIT, Cambridge, MA, 1988.

Heckerman, D. E. (1990) Probabilistic Similarity Networks, Ph.D. dissertation, MIS Department, Stanford University, Stanford, Ca.

Howard, R. A. (1990) "From Influence to Relevance to Knowledge," in Influence Diagrams, Belief Nets and Decision Analysis, Oliver, R. and Smith, J. (Eds.), New York: John Wiley & Sons.

Levis, A. H. (1992) "A Colored Petri Net Model of Command and Control Nodes," in Command, Control and Communications: Advanced Concepts and Paradigms, Jones, C.R. (ed.) AIAA.

Levis A. H., (1995). Human Interaction with Decision Aids: A Mathematical Approach, W. B. Rouse Editor, *Human / Technology Interaction in Complex Systems*, pp 85-171. JAI Press.

Monguillet J.-M. and Levis A. H., 1991. Modeling and Evaluation of Variable Structure Command and Control Organizations, C. R. Jones Editor. *Toward a Science of Command and Control , and Communications*, Progress in Astronautics and Aeronautics. **156**, pp. 193-219. AIAA, Washington, DC. 1991.

Neapolitan, R.E. (1990) Probabilistic Reasoning in Expert Systems: Theory and Algorithms, New York: Wiley.

Pearl, J. (1988) Probabilistic Reasoning In Intelligent Systems: Networks of Plausible Inference, San Mateo, CA: Morgan Kaufmann.

Pete, A., Pattipati, K.R., and Kleinman, D.L. (1995) Designing Organizations with Congruent Structures, Proceedings of the First International Symposium on Command and Control Research and Technology, Washington D.C., pp. 197-209.

Pete, A., Pattipati, K.R., and Kleinman, D.L. (forthcoming) Optimization of Decision Networks in Structured Task Environments, IEEE Transactions on Systems, Man and Cybernetics.

Shachter, R. D., (1986) "Evaluating Influence Diagrams," Operations Research, Vol. 34, 871-882.

Shachter, R. D., (1990) "An Ordered Examination of Influence Diagrams," Networks, Vol. 20, 535-563.

Tatman, J.A. and Shachter, R.D. (1990) "Dynamic Programming and Influence Diagrams", IEEE Transactions on Systems, Man and Cybernetics, Vol. 20, 2, pp. 365-379.

Zaidi S. A. K., 1994. *Validation and Verification of Decision Making Rules*. Ph. D. Dissertation. Center of Excellencer in Command, Control, Communications and Intelligence, George Mason University, Fairfax, VA, 1994.

5. MEETINGS

Dr. Levis visited the the Naval Postgraduate School, Monterey, CA in August 1995 and briefed the staff on GMU research. Dr. Levis and Mr. Perdu attended the semi annual review of the A2C2 program at the Naval Postgraduate School, Montery, CA to brief the team on GMU research.

Drs. Levis and Buede and Messrs. Perdu and Wagenhals attended the 1996 Symposium on C2 Research and Technology at the naval Postgraduate School in Monterey, CA and presented five papers deruved from this research.

Dr. Levis and Mr. Perdu attended the TADMUS program review held at GMU.

6. CHANGES

Changes in the scope of work of this project, as a result of two contract modifications, have been documented in section 2.

7. CURRENT PERSONNEL

Prof. Alexander H. Levis	Principal Investigator
Prof. Dennis Buede	
Dr. Abbas K. Zaidi	Consultant
Mr. Lee Wagenhals	Research Instructor (Ph.D.)
Mr. Didier Perdu	Research Instructor (Ph.D.)
Mr. Eric Tsibertzopoulos	Graduate Research Asst. (MS)
Ms. Etsiwohot Dinka	Undergraduate Research Assistant

8. DOCUMENTATION

1. Hedy L. Rashba (1993). Problems in Concurrency and Coordination in Decision Making Organizations. Report GMU/C3I-143-R, C3I Center, George Mason University, Fairfax, VA.

2. A. H. Levis (1993). Adaptive Decision Making and Coordination in Variable Structure Organizations. Report GMU/C3I-147-IR, C3I Center, George Mason University, Fairfax, VA.

3. Zhenyi Jin (1994). Deadlock and Trap Analysis in Petri Nets. MS Thesis, Systems Engineering Department, George Mason University, Fairfax, VA.

4. A. H. Levis and D. M. Perdu (1994). Object Oriented Design of Decision Making Organizations, *Proc. 1994 The First Workshop on Command Informations Systems,* Oxon, UK. pp. 372-384. Defence Research Agency. Malvern.

5. A. K. Zaidi (1994). Validation and Verification of Decision Making Rules, Report GMU/C3I-155-TH, C3I Center, George Mason University, Fairfax, VA.

6. A. H. Levis and D. M. Perdu (1994). Object Oriented Design of Decision Making Organizations, A. H. Levis & I. S. Levis (Editors), *The Science of Command and Control: Part III, Coping with Change,* AFCEA International Press, Fairfax, VA.

7. A. H. Levis, D. M. Perdu and A. K. Zaidi (1994). Adaptive Decision Making and Coordination in Variable Structure Organizations. Report GMU/C3I-151-IR, C3I Center, George Mason University, Fairfax, VA.

8. A. K. Zaidi and A. H. Levis (1995). Rule Decomposition and Validation for Distributed Decision Making, *Proc. 1995 First International Symposium on Command and Control Research and Technology,* National Defense University, Washington, DC. pp. 210-217.

9. A. K. Zaidi and A. H. Levis (1995). Validation and Verification of Decision Making Rules, *Proc. 1995 6th IFAC/IFIP/IFORS/IEA Symposium on Analysis, Design and Evaluation of Man-Machine Systems,* MIT, Cambridge, MA. pp. 53-60.

10. A. H. Levis, D. M. Perdu & A. K. Zaidi (1995). CAESAR II: Computer Aided Evaluation Of System Architectures, Report GMU/C3I-162-R, C3I Center, George Mason University, Fairfax, VA.

11. A. K. Zaidi and A. H. Levis (1995). Object Oriented Design of a Multilevel Hierarchical Organization Structure, *Proc. US/Portugal Workshop On Underwater Vehicles and Intelligent Control,* Lisbon, Portugal. University of So. Louisianna Press.

12. A. K. Zaidi and A. H. Levis (1995).Validation and Verification of Decision Making Rules. Accepted for publication in *Automatica.*

13. A. K. Zaidi and A. H. Levis (1996). On Generating DIS Architectures Using Genetic Algorithms, Paper GMU/C3I-166-P, C3I Center, George Mason University, Fairfax, VA.

14. Didier M. Perdu and Alexander H. Levis (1996). Object Oriented Design of An Air Combat Operations System Using Eagle Vision, *Proc. of the 1996 C2 Symposium,* Monterey, California, June 24 - 28.

15. Didier M. Perdu and Alexander H. Levis (1996). Distributed Process Coordination in Adaptive Command and Control Teams, *Proc. of the 1996 C2 Symposium,* Monterey, California, June 24 - 28.

16. Dennis M. Buede and Lee W. Wagenhals (1996). Influence Diagram Representation of Dynamic, Distributed Decision Making, *Proc. of the 1996 C2 Research and Technology Symposium,* Monterey, California, June 24 - 28.

17. Lee W. Wagenhals (1996). Digitization of the Battlefield: Approaches for Assessing Behavior and Performance of Distributed Command and Control Systems, *Proc. of the 1996 C2 Research and Technology Symposium,* Monterey, California, June 24 - 28.

18. Didier Leroy and Didier Hoffman (1996). Criteria of C3I Effectiveness, *Proc. of the 1996 C2 Research and Technology Symposium,* Monterey, California, June 24 - 28.